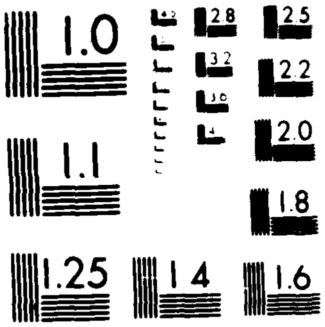
AD-A179 356   ADAPTIVE BEAMFORMING WITH THE TRANSFORM DOMAIN LMS   1/1
(LEAST MEAN-SQUARE)(U) ILLINOIS UNIV AT URBANA COLL OF
ENGINEERING K Z SIEJKO FEB 87 UILU-ENG-87-2214

UNCLASSIFIED   N00014-84-C-0149   F/G 9/3   NL



MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU OF STANDARDS-1963-A

February 1987 UILU-ENG-87-2214

COORDINATED SCIENCE LABORATORY
College of Engineering

2

ADAPTIVE BEAMFORMING WITH THE TRANSFORM DOMAIN LMS ALGORITHM

Krzysztof Z. Siejko

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

87 3

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION
Unclassified

1b. RESTRICTIVE MARKINGS
None

A179356

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE

3 DISTRIBUTION / AVAILABILITY OF REPORT
Approved for public release;
distribution unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
UILU-ENG-87-2214

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
Coordinated Science Lab
University of Illinois

6b OFFICE SYMBOL *(If applicable)*
N/A

7a. NAME OF MONITORING ORGANIZATION
Office of Naval Research

6c. ADDRESS *(City, State, and ZIP Code)*
1101 W. Springfield Ave.
Urbana, IL 61801

7b. ADDRESS *(City, State, and ZIP Code)*
800 N. Quincy St.
Arlington, VA 22217

8a. NAME OF FUNDING / SPONSORING ORGANIZATION
Joint Services Electronics Program

8b. OFFICE SYMBOL *(If applicable)*

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
N00014-84-C-0149

8c. ADDRESS *(City, State, and ZIP Code)*
800 N. Quincy St.
Arlington, VA 22217

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

11. TITLE *(Include Security Classification)*
Adaptive Beamforming with the Transform Domain LMS Algorithm

12 PERSONAL AUTHOR(S)
Siejko, Krzysztof Z.

13a. TYPE OF REPORT
Technical

13b. TIME COVERED
FROM ________ TO ________

14. DATE OF REPORT *(Year, Month, Day)*
February 1987

15. PAGE COUNT
70

16 SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
adaptive beamforming, array processing, transform domain LMS algorithm

19 ABSTRACT *(Continue on reverse if necessary and identify by block number)*

In adaptive digital filtering, the Transform Domain LMS algorithm (TRLMS) has shown the ability of improving the convergence rate of the Widrow-LMS algorithm at the expense of implementing a fixed orthogonal transform. In broadband beamforming, each sensor of an array is typically followed by a tapped delay line to provide frequency-dependent array weighting. The weights can be adjusted adaptively to steer the nulls of the beampattern toward any undesired sources. For Least Mean-Square (LMS) based adaptive arrays in a multisignal environment, convergence is highly data dependent and is characterized by highly disparate modes, resulting in slow and noisy adaptation. This work examines the feasibility of extending the TRLMS algorithm to the multichannel scenario of wideband beamforming for the purpose of improving convergence rates.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT
☒ UNCLASSIFIED/UNLIMITED □ SAME AS RPT. □ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL

22b. TELEPHONE *(Include Area Code)*

22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR

83 APR edition may be used until exhausted.
All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

ADAPTIVE BEAMFORMING WITH THE TRANSFORM DOMAIN LMS ALGORITHM

BY

KRZYSZTOF Z. SIEJKO

B.S., University of Illinois, 1986

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1987

Urbana, Illinois

ABSTRACT

In adaptive digital filtering, the Transform Domain LMS algorithm (TRLMS) has shown the ability of improving the convergence rate of the Widrow-LMS algorithm at the expense of implementing a fixed orthogonal transform. In broadband beamforming, each sensor of an array is typically followed by a tapped delay line to provide frequency-dependent array weighting. The weights can be adjusted adaptively to steer the nulls of the beampattern toward any undesired sources. For Least Mean-Square (LMS) based adaptive arrays in a multisignal environment, convergence is highly data dependent and is characterized by highly disparate modes, resulting in slow and noisy adaptation. This work examines the feasibility of extending the TRLMS algorithm to the multichannel scenario of wideband beamforming for the purpose of improving convergence rates.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Professor W.K. Jenkins for his support and interest during the writing of this thesis. I greatly appreciate his time and insight, and the opportunity to be introduced to the exciting topic of adaptive signal processing. Also, I would like to express a special thanks to Regency Systems, Inc. for allowing me the use of their computer equipment and facilities. And my deepest appreciation goes to Barbara, for her great patience, support, and understanding during my entire graduate program.

TABLE OF CONTENTS

CHAPTER PAGE

CHAPTER 1

INTRODUCTION

In the area of adaptive filtering, the Transform Domain LMS algorithm (TRLMS) has shown the ability of improving the convergence rate of the Widrow-LMS algorithm at the expense of implementing a fixed orthogonal transform [1], [2]. In broadband beamforming, each sensor of an array is typically followed by a tapped delay line or digital filter to provide the necessary frequency dependent array weighting. The weights can be adjusted adaptively to steer the nulls of the beampattern toward any undesired sources. For Least Mean-Square (LMS) based adaptive arrays in a multisignal environment, convergence is highly data dependent and is characterized by highly disparate modes. This results in very slow and noisy adaptation.

Much attention has been given to fast adaptive arrays. The well-studied Compton Loop offers a continuous time solution [3], [4], and fast algorithms more robust than the LMS are known to exist [5]. Orthogonalization methods that employ the adaptive multichannel lattice filter [6], [7] and eigenvector preprocessors [8] have also been proposed to improve the convergence rates.

This thesis examines the feasibility of extending the TRLMS algorithm to the multichannel scenario of wideband beamforming for the purpose of improving convergence rates. A special class of beamformer, the Generalized Sidelobe Canceller, is specifically considered. This structure converts the problem of linearly constrained adaptive beamforming to one of unconstrained multiple reference noise cancelling. Applying the TRLMS algorithm to this multiple input case is nontrivial, as it will be shown that many different arrangements of the input data vectors are possible. In general, the performance of a fixed transform algorithm is data dependent [1]. The intent here is not to develop means by which to choose a transform, but to provide an introduction to this topic and to determine if the use of a fixed transform can be justified in this highly data dependent scenario.

Studies of the performance of the Generalized Sidelobe Canceller based on signal environment and on the type of transform chosen will be verified by computer. It will be shown that the TRLMS algorithm can be effective in some situations. Performance appears to be dependent upon many factors, including the number of sensors, the number of jammers and their angles of incidence, and the arrangement of the data vectors.

CHAPTER 2
BACKGROUND

This two-part discussion presents the necessary background in linearly constrained adaptive beamforming and transform domain adaptive FIR filtering. These seemingly unrelated topics will be first discussed separately, and then later joined to apply the ideas of transform domain LMS adaptation to improve performance of the Generalized Sidelobe Canceller, sometimes called the Griffiths-Jim beamformer. The beamforming application doubles as an experimental base in which to verify the extension of the said adaptive filter to multiple inputs.

Unless noted otherwise, assume that all signals are zero mean and wide-sense stationary, and that all wavefronts are planar (generated by a far field source). Assume all antenna elements are ideal and omnidirectional, with array geometries linear and uniformly spaced at distance d. Discretized data are assumed to be available at each element.

2.1 Adaptive Beamforming

The beamforming problem involves selecting a desired propagating source (signal) in an environment where other undesired directional jammers or noises may be present. Adaptive beamforming involves tracking signal parameters which are time varying or unknown. Reference [9] contains a fine introduction to this topic.

In the conventional narrowband signal case, each sensor of the linear array is weighted by a complex weight and summed to produce the output, as shown in Fig. 2.1a. The magnitude and phase of each weight can be adjusted adaptively to achieve maximum signal-to-interference ratio (SIR), or calculated directly if the signal parameters are known [9]. One sensor is commonly used as a fixed reference to prevent the output from decaying to zero during adaptation.

A linear array will have equivalent response for angles symmetric about its axis (line formed by the sensors), whereas a circular arrangment can uniquely respond to all 360°. To avoid aliasing due to spatial sampling by the sensors, the sampling frequency (array spacing) 1/d must be at least twice the spatial frequency of the wave, $1/\lambda$. Therefore, $d \leq \lambda /2$. An array of K sensors has the capability to reject up to K-1 jammers and pass one desired signal. This is rationalized by envisioning the solution of a linear system of K equations and K unknowns.

For broadband signals, each weight in Fig. 2.1a. is replaced by a tapped delay line, as shown in Fig. 2.1b, to provide the frequency dependent magnitude and phase shift over the operating bandwidth of the array. The weights can be adapted using many different algorithms. Also, several array configurations can be used [9]. An important point is that adapting the weights with a simple algorithm such as LMS would result in their decaying to zero. This corresponds to a minimization of the output, which happens to be zero in this case. As mentioned, an unweighted reference sensor can be used to prevent this, causing a useful output to be present even if the weights became zero. However, the desired signal may still be cancelled if its relative power level is high, similar to behavior encountered in adaptive noise cancelling when desired signal components are present in the reference channel [9].

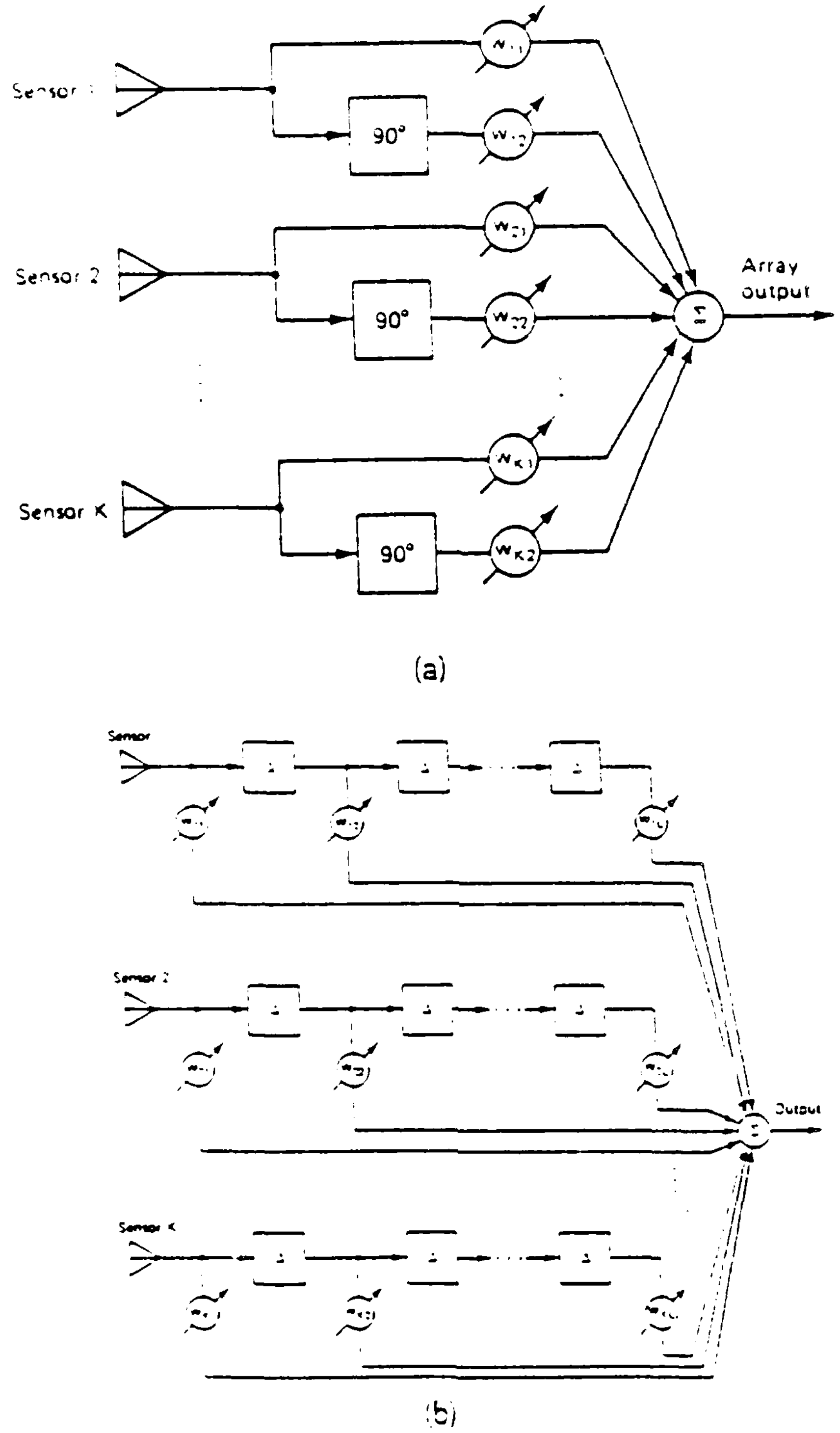


Fig. 2.1 (a) Conventional narrowband adaptive array and its (b) broadband counterpart. From S. Stearns and B. Widrow, Adaptive Signal Processing, © 1985, Prentice Hall.

One solution proposed by Frost [10] places a linear constraint on the weights to prevent their decay to zero. This constraint is also used to process the desired signal, fixing its response regardless of signal-to-interference ratio (SIR). Another similar linearly constrained adaptive broadband array will be the focus of attention, namely the Generalized Sidelobe Canceller.

2.1.1 The Generalized Sidelobe Canceller

Figure 2.2 shows the Generalized Sidelobe Canceller Broadband Beamformer (GSC), first analyzed by Griffiths and Jim [11]. This structure has been of recent interest to researchers due to its flexibility and simplicity [9], [12]. It is appealing in the sense that any unconstrained adaptive algorithm or filter structure $W_k(n)$ may be used for its implementation. For example, Reference [12] reports improved SIR for the GSC using pole-zero filters in place of conventional tapped delay lines, although only the FIR case will be considered here. The only constraint on the GSC is that the location of the desired signal must be known a priori.

The GSC has a direct analogy to adaptive noise cancelling with multiple reference inputs, making it easy to analyze and understand. Refer to Fig. 2.2. The desired broadband signal $S_d(n)$ impinges upon the array from the look-direction, defined as the angle $\theta = 0^{\circ}$ with respect to the array normal. Appropriate steering delays at each sensor may be used for alignment if necessary. Since $S_d(n)$ is in phase at each sensor, it is blocked from the filter inputs $X_k(n)$ by the subtractive preprocessing stage and rerouted to the "primary input" channel $d(n)$. Thus, $S_d(n)$ passes to the array output unaltered, and additional processing may done by inserting a fixed filter

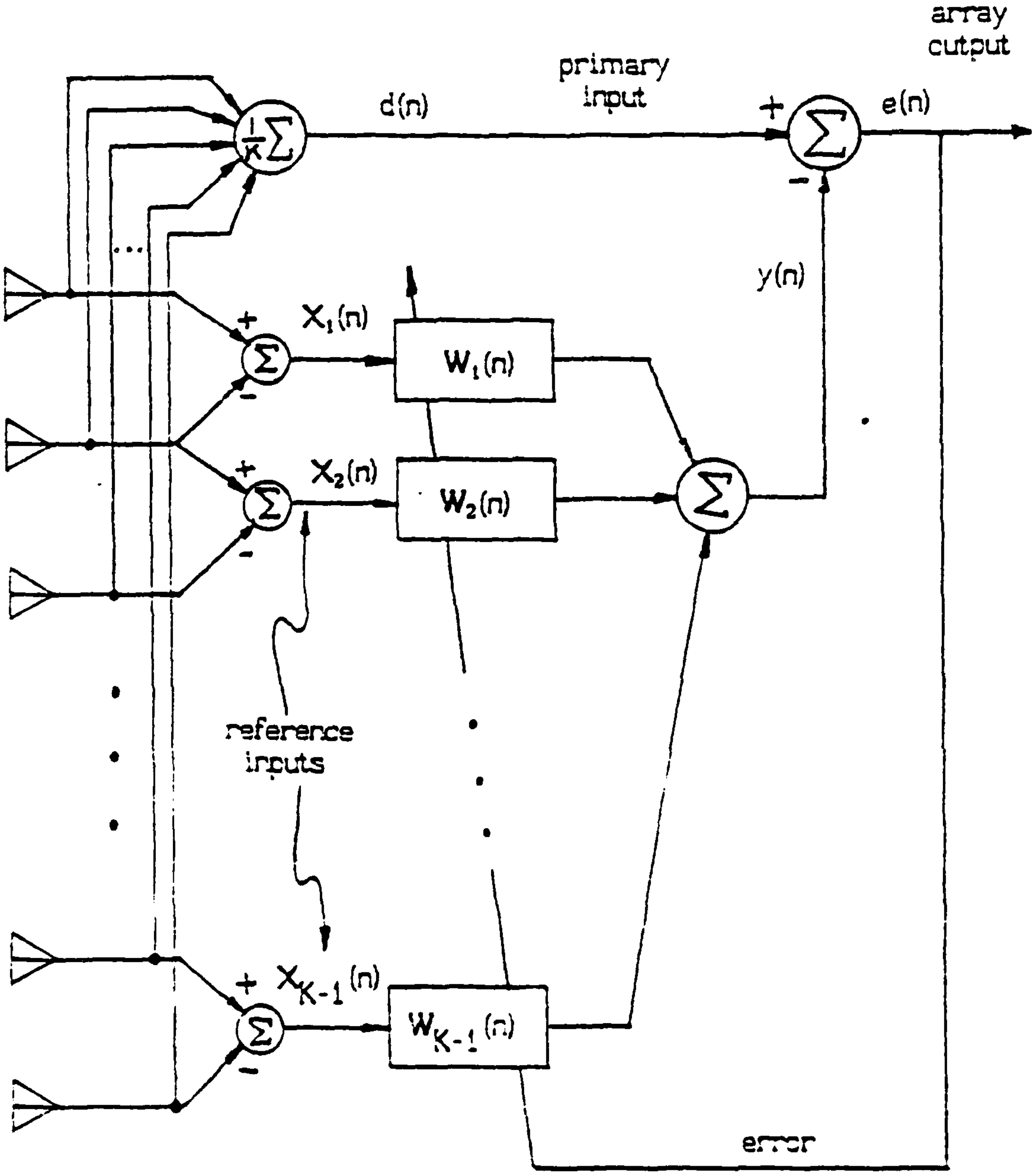


Fig. 2.2 The Generalized Sidelobe Canceller Broadband Beamformer (GSC).

in the d(n) channel. Any other broadband signals $S_i(n)$ (assumed to be uncorrelated with $S_d(n)$) arriving at angles off the look-direction will have components in both d(n) and the "reference input" channels $X_k(n)$, $1 \leq k \leq K-1$. These components, although correlated, differ due to effects of the preprocessor and the spatial sampling process of the array. For example, the contribution of $S_i(n)$ to d(n) is

$$N_i^0(n) = \frac{1}{K}\sum_{k=1}^{K} S_i[nT-(k-1)\Delta_i+\phi_i] \qquad (2.1)$$

with arbitrary phase ϕ_i and intersensor propagation delay

$$\Delta = \frac{2\pi d}{\omega\lambda T}\sin\theta \quad , \qquad (2.2)$$

determined by angle θ, sensor spacing d, radian frequency ω and its corresponding wavelength λ (for a nondispersive media, $\omega\lambda$= constant), and sampling period T. The contribution of $S_i(n)$ to the k^{th} reference input is

$$N_{i_k}(n) = S_i[nT-(k-1)\Delta_i+\phi_i] - S_i[nT-k\Delta_i+\phi_i] \quad . \qquad (2.3)$$

The array output e(n)=d(n)-y(n) serves as the error signal to be minimized by the adaptive filters of impulse response $W_k(n)$. Let $X_k(n)$ be a vector of length corresponding to $W_k(n)$. Then,

$$e(n) = d(n) - \sum_{k=1}^{K-1}[X_k(n)^t W_k(n)] \quad . \qquad (2.4)$$

For p jamming signals ($1 \leq p \leq K-1$), let

$$N^{o}(n) = \sum_{i=1}^{p} N^{0}_{i}(n) \qquad (2.5a)$$

and

$$N^{1}_{k}(n) = \sum_{i=1}^{p} N_{i_k}(n) \quad . \qquad (2.5b)$$

The classic adaptive noise cancelling scenario [13] is now clear; the primary input signal d(n) consists of a desired signal $S_d(n)$ plus a noise component N^{0}, and the kth reference input consists of noise N^{1}_{k}, which is correlated to N^{0} but not to $S_d(n)$. The weights of the W_k's are adaptively adjusted to cancel (2.5a) with the multiple references of (2.5b), minimizing e(n) in the mean-square sense while leaving $S_d(n)$ intact.

The GSC can be categorized [9] as a "hard constraint" structure, so called because it maintains a fixed look-direction response regardless of signal strengths. Other predecessor structures with "soft constraints" make use of training signals or require a minimum SIR to maintain a specified look-direction sensitivity. Other classifications are "linearly constrained minimum variance beamformer" or "maximum likelihood array," which again imply that the output power of the array can be minimized without fear of cancelling any useful output (assuming uncorrelated inputs). A point worth noting is that as the desired signal deviates from the look-direction, its wavefront components become out of phase at each sensor. This signal then "leaks" into the reference channels, since the preprocessor can no longer completely block it. If the relative power of this signal is high, the weights would adapt to cancel $S_d(n)$ from the output, due to correlated components in the primary and reference channels. This property is useful in target bearing estimation applications [9].

Gooch and Shynk [12] have shown that the optimal solution for $W_k(n)$ contains both poles and zeros. This solution will perfectly cancel any coherent interference. The transfer function, however, contains unrealizable fractional powers of Z (Z-Transform variable) in the denominator which correspond to Δ in (2.2). This implies that an effective FIR solution will most likely require a very large filter order to approximate the impulse response of the IIR solution.

2.1.2 Wiener Solution for the GSC

Refer to Fig. 2.3. Note that the sensors and preprocessor stage have been removed for simplicity. For a K sensor array there are K-1 reference inputs. Let L be the length of the FIR filter at each reference, represented by the weight vector

$$W_k(n) = [\ w_{ko}(n) \ \ w_{k1}(n) \ \ \ldots \ \ w_{k(L-1)}(n)\ ]^t \qquad (2.6a)$$

and its corresponding data vector

$$X_k(n) = [\ x_{ko}(n) \ \ x_{k1}(n) \ \ \ldots \ x_{k(L-1)}(n)\ ]^t \quad . \qquad (2.6b)$$

Since the filter bank output y(n) consists of the summation of all (K-1)L weighted outputs, we can arrange the (K-1)L dimensional weight and data vectors $\overline{W}_n$ and $\overline{X}_n$ to produce

$$y(n) = \overline{W}_n^{\,t}\,\overline{X}_n = \overline{X}_n^{\,t}\,\overline{W}_n \quad , \qquad (2.7)$$

where the overbar denotes a system vector. The ordering of the elements x_{kl} and w_{kl}

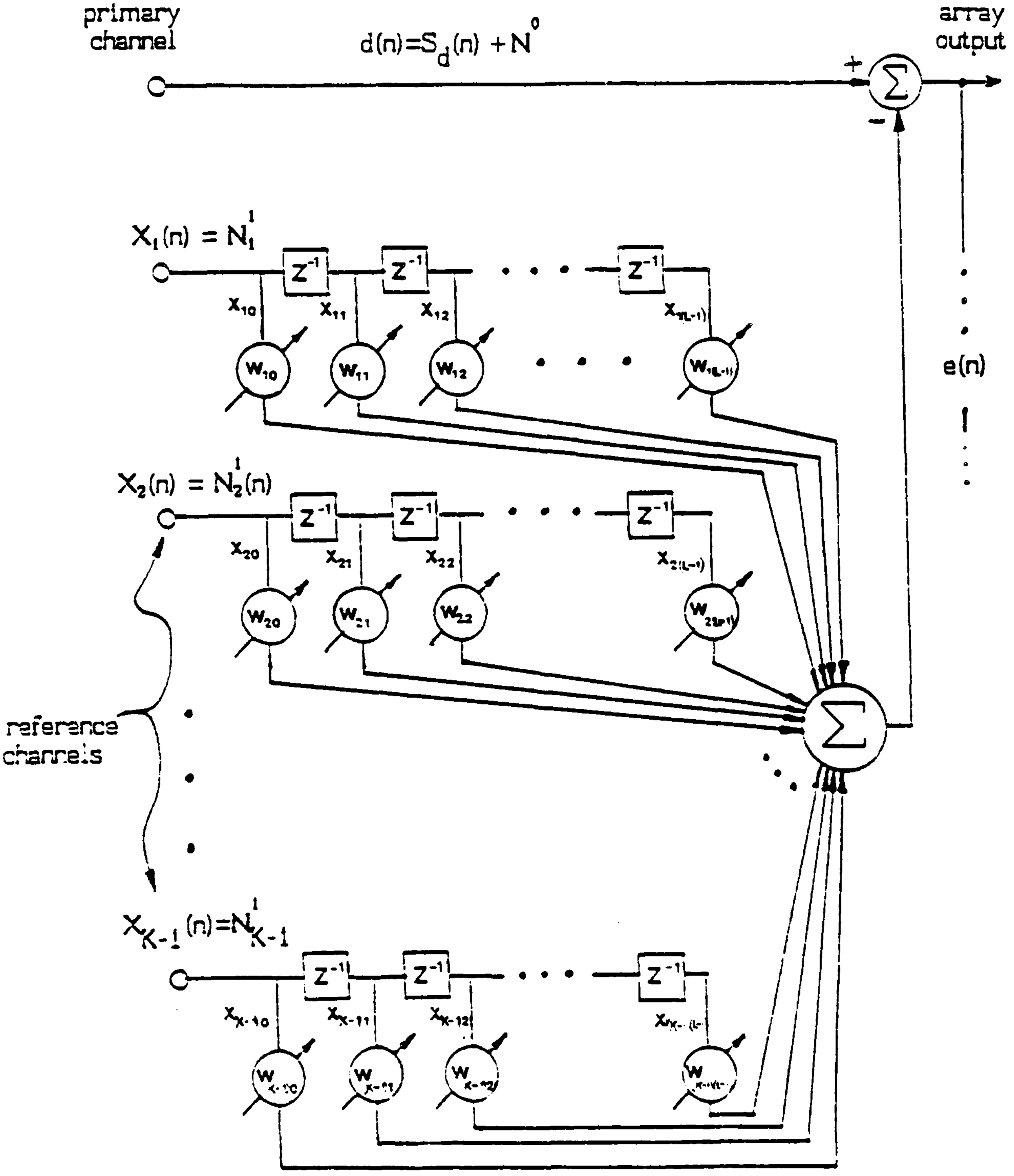


Fig. 2.3 Multiple reference noise canceller analogy to the GSC. The sensors and preprocessing stages have been removed.

is arbitrary, an important point to be considered further in Chapter 4. For this analysis, $\overline{W}_n$ and $\overline{X}_n$ can be arranged by "stacking" the subvectors of (2.6) as

$$\overline{W}_n = \begin{bmatrix} W_1(n) \\ W_2(n) \\ \vdots \\ W_{K-1}(n) \end{bmatrix} \quad \text{and} \quad \overline{X}_n = \begin{bmatrix} X_1(n) \\ X_2(n) \\ \vdots \\ X_{K-1}(n) \end{bmatrix} \quad . \tag{2.8}$$

The remainder of the analysis is identical to that for the standard discrete Wiener solution [9]. The quantity to be LMS minimized is e(n)=d(n)-y(n). Taking $E[e(n)^2]$ yields

$$\begin{aligned} E[e(n)^2] &= E[d(n)^2] - 2E[d(n)\overline{W}_n^{\,t}\overline{X}_n] + E[\overline{W}_n^{\,t}\overline{X}_n\overline{X}_n^{\,t}\overline{W}_n] \\ &= E[d(n)^2] - 2\overline{W}_n^{\,t}P_x + \overline{W}_n^{\,t}R_x\overline{W}_n \end{aligned} \tag{2.9}$$

from which the autocorrelation matrix $R_x = E[\,\overline{X}_n\overline{X}_n^{\,t}]$ and the cross-correlation vector $P_x = E[\,d(n)\overline{X}_n\,]$ are defined. The optimal Wiener solution is achieved by taking partials of (2.9) with respect to w_{kl} as

$$\frac{\partial E[e(n)^2]}{\partial w_{kl}} = 0 \qquad \begin{aligned} k &= 1,2,\ldots,K-1 \\ l &= 0,1,\ldots,L-1 \end{aligned} \tag{2.10}$$

to yield the familiar

$$\overline{W}_{opt} = R_x^{-1}P_x \quad . \tag{2.11}$$

Many algorithms such as Widrow's LMS or the RLS [14] exist to solve (2.11) iteratively, and the flexibility of the GSC allows any suitable choice. One choice which shows promise of improved convergence rate over the LMS algorithm is the Transform Domain LMS algorithm. The next section examines this algorithm for the general case.

2.2 Transform Domain Adaptive Digital Filters

It is known that the convergence rate of an LMS adaptive filter is governed by the eigenvalues of the input covariance matrix, which in turn are related to the power spectrum of the input [2]. The idea behind transform domain adaptive filtering is simply to whiten the input power spectrum. This equalizes the eigenvalues, allowing the "modes" of the system to converge at the same rate. The beauty of the TRLMS filter lies in the fact that no inverse transform is necessary; the transform acts only as an orthogonal mapping of the weight vector W_n and the data vector X_n into a domain where the desired power normalization is feasible.

The Widrow LMS algorithm is given by

$$W_{n+1} = W_n + 2\mu e_n X_n . \tag{2.12}$$

An expression for the convergence of the mean weight-error vector is

$$E[V_n] = [I - 2\mu R_x]^n V_o , \tag{2.13}$$

where $V_n = W_n - W_{opt}$, R_x is the autocorrelation matrix (assumed to be stationary), and I is the identity matrix [9]. Equation (2.13) can be decoupled to its modal form by applying the modal matrix Q such that

$$V_n' = Q^t V_n \quad \text{and} \quad R_x = Q \Lambda Q^t \quad , \tag{2.14}$$

where Q is a matrix whose columns are the orthonormal eigenvectors of Rx, and Λ is the diagonal matrix of their corresponding eigenvalues, λ_i. For simplicity, assume that a distinct eigenvector exists for each eigenvalue. Combining (2.13) and (2.14) yields

$$E[V_n'] = [I - 2\mu\Lambda]^n V_o' \tag{2.15}$$

which clearly can be separated into N modal equations. The time constant for the i^{th} mode, calculated from the geometric ratio $1\text{-}2\mu\lambda_i$, is shown to be [9]

$$\tau_i = 1/4\mu\lambda_i. \tag{2.16}$$

Clearly, overall convergence is governed by the mode corresponding to the smallest eigenvalue, λ_{min}. A highly disparate eigenvalue spread will result in poor performance. Since the eigenvalues of R_x are bounded by the minimum and maximum values of the input power spectrum [15], whitening of the input results in all eigenvalues being identical. For fixed levels of steady-state MSE, λ_{min} then becomes maximum and convergence is improved [2]. An orthogonal transform permits some direct manipulation of the eigenvalues, the degree of which depends on the transform.

The TRLMS filter is shown in Fig. 2.4. For each time index n, the data vector X_n is transformed by an orthogonal N ×N matrix T. Defining $y(n)$, $\underline{\overline{\omega}}_n$, and S_n to be

$$y(n) = S_n^t \underline{\overline{\omega}}_n = \underline{\overline{\omega}}_n^t S_n \, , \tag{2.17}$$

$$S_n = TX_n \quad , \tag{2.18}$$

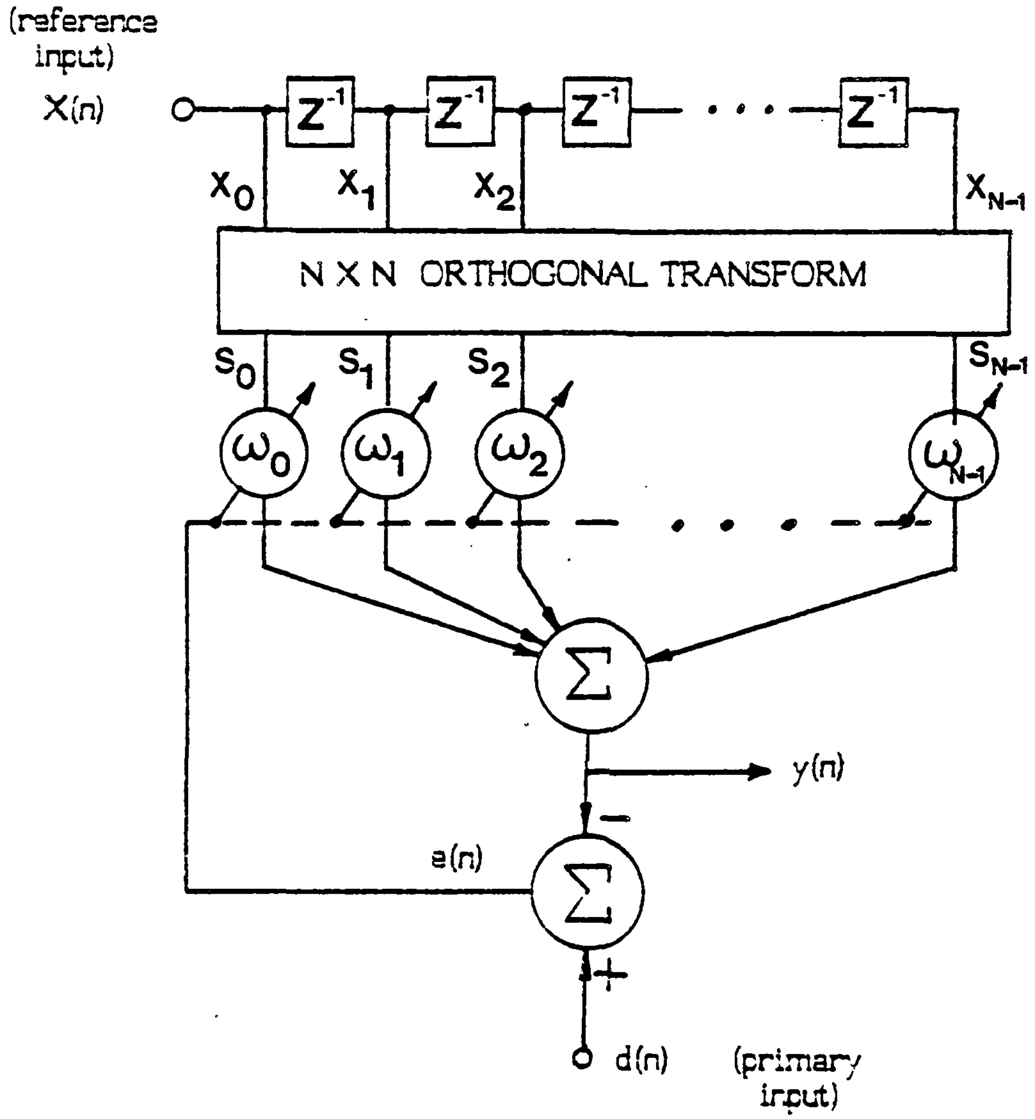


Fig. 2.4 The transform domain adaptive filter.

the optimum Wiener solution can be shown to be

$$\overline{\underline{\omega}}_{opt} = R_s^{-1} P_s \quad , \tag{2.19a}$$

where

$$R_s = E[S_n^* S_n^t] \quad \text{and} \quad P_s = E[d(n)\, S_n^*] \; . \tag{2.19b}$$

The relationship between time and transform domain solutions can be established. From (2.18) and (2.19) we have

$$R_s = T^* R_x T^t \qquad \text{and} \qquad P_s = T^* P_x \quad , \tag{2.20}$$

and substituting into (2.19a) yields

$$\overline{\underline{\omega}}_{opt} = (T^* R_x T^t)^{-1} T^* P_x = (T^t)^{-1} R_x^{-1} P_x = (T^t)^{-1} W_{opt} . \tag{2.21}$$

Thus the transform maps the weight vector as well, and (2.21) can be used to recover the time domain solution if neseccary.

As discussed in [2], the minimum mean-square errors of the time domain and the transform domain filters will be identical as long as T is orthogonal or,

$$T T^{*t} = \kappa I \; , \tag{2.22}$$

where κ is a scaling factor dependent upon N and type of transform. For real transforms, $T^{*t} = T^t = T^{-1}$. These points illustrate the fact that no inverse transform is needed.

The Transform Domain LMS algorithm [2] is given by

$$\underline{\bar{\omega}}_{n+1} = \underline{\bar{\omega}}_{n} + \eta \, S_{n}^{*} e_{n} \tag{2.23}$$

where η is a constant convergence factor. Similar to (2.13), the mean weight-error vector is

$$E[\, \bar{\nu}_{n} \,] = [\, I - 2\eta R_{s} \,]^{n} \bar{\nu}_{o} \tag{2.24}$$

where $\underline{\nu}_{n} = \underline{\bar{\omega}}_{n} - \underline{\bar{\omega}}_{opt}$. It is known that unitary transforms preserve the variance, and in turn, the eigenvalues of an autocorrelation matrix. Therefore, with appropriate selection of η, (2.24) and (2.13) will have the same time constants. By setting $\eta = \mu/\kappa$, where κ is N, N/2, and N for the DFT, DCT, and DHT respectively, identical performance of the LMS and the TRLMS filters is expected [2].

2.2.1 Orthogonal Transforms

Many discrete orthogonal transforms exist. The three which will be considered here are the Discrete Fourier Transform (DFT), the Discrete Hadamard Transform (DHT), and the Discrete Cosine Transform (DCT). The optimal Karhunen-Loeve Transform (KLT) will perfectly decorrelate an input and diagonalize its autocorrelation matrix. The KLT is based on input statistics and generally not used in an adaptive environment where signal parameters are time varying. Identical to (2.14), it involves solving for the eigenvectors of $R_{\hat{x}}$, and no general fast algorithm exists. The KLT serves merely as a reference for the suboptimal cases.

The DFT is ideal for complex valued inputs and fast VLSI implementations exist, but for real inputs the complex arithmetic is cumbersome. The basis functions for a matrix representation are the familiar complex exponentials,

$$t_{l,m} = e^{-j2\pi lm/N}, \qquad 0 \leq l,m \leq N\text{-}1. \tag{2.25}$$

The DCT, a similar counterpart of the DFT, uses only real coefficients, and fast algorithms exist for a limited number of values for N [16]. Its kernel is of the form

$$\begin{aligned} t_{l,m} &= \alpha(l)\cos\left\{\frac{\pi l(2m+1)}{2N}\right\}, \qquad 0 \leq l,m \leq N\text{-}1, \\ \alpha(0) &= \frac{1}{\sqrt{2}}, \quad \alpha(l) = 1 \quad l \neq 0. \end{aligned} \tag{2.26}$$

A 2N FFT can be used for computation of (2.26) due to the similarities between them. The DCT has shown performance closer to the optimal Karhunen-Loeve Transform (KLT) than other transforms for many applications, including image coding and channel equalization, and is asymptotically equivalent to the KLT for Markov-1 signals as N → ∞ [16].

The DHT has been of great interest due to its computational simplicity. Composed of Walsh basis functions which take on values of +1 and -1, the DHT can be computed with simple additions and subtractions. For example, the DHT matrix for N=4 is

$$T_{DHT} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}. \tag{2.27}$$

and it has the special property that $T = T^{-1} = T^t$.

The identity matrix can also be considered as an orthogonal transform. Since the I matrix simply passes inputs to the weights unaltered, use of this transform in the TRLMS algorithm is equivalent to the LMS algorithm. This fact will simplify comparisons of the TRLMS algorithm to the LMS case.

2.2.2 Self-Orthogonalizing TRLMS Adaptive Filter

Consider now the self-orthogonalizing TRLMS algorithm [2], where

$$\underline{\bar{\omega}}_{n+1} = \underline{\bar{\omega}}_n + \mu \hat{R}_s^{-1} S_n^* e_n \tag{2.28}$$

and

$$E[\bar{\nu}_n] = [I - 2\mu \hat{R}_s^{-1} R_s]^n \bar{\nu}_0 \quad . \tag{2.29}$$

$\hat{R}_s$ is a diagonal matrix whose elements are the power estimates of each s_i, namely,

$$\hat{R}_s = \text{diag}[\ \hat{\sigma}_0^2 \quad \hat{\sigma}_1^2 \cdots \hat{\sigma}_i^2 \cdots \hat{\sigma}^2_{N-1}\ ]. \tag{2.30}$$

It is known that the transforms under consideration will approximately diagonalize the R_x matrix. For the optimal KLT case the diagonal elements of R_s will be equivalent to the input variances [17], and $\hat{R}_s^{-1} R_s = I$. Eq. (2.29) would then be truly decoupled as

$$E[\nu_i(n)] = [1 - 2\mu]^n \nu_i(0) \quad . \tag{2.31}$$

with all modes converging with time constant $\tau_i = 1/4\mu$. For the other suboptimal cases, the residual off-diagonal elements of R_s cause cross coupling of the modal equations, resulting in some disparity among the time constants. The effectiveness of a specific transform is strongly dependent upon the class of inputs. This is illustrated experimentally in [1].

An effective method of estimating the power $\sigma_i^2(n)$ is via a single pole low-pass filter as given in [2]:

$$\hat{\sigma}_i^2(n) = \beta\,\hat{\sigma}_i^2(n-1) + (1-\beta)\,s_i(n)\,s_i(n)^* \,, \tag{2.32}$$

where β is the smoothing constant which controls estimation accuracy and tracking ability. Reference [2] provides a detailed discussion on the effects of choices of β and initial power estimate $\sigma_i^2(0)$ on the convergence rate.

Thus, the self-orthogonalizing TRLMS algorithm is identical to (2.23) with a time varying convergence factor, namely $\eta=\mu/\hat{\sigma}_i^2(n)$. The algorithm to be examined in this thesis is

$$\omega_i(n+1) = \omega_i(n) + \frac{2\mu}{\hat{\sigma}_i^2(n)}\, e(n)\, s_i(n)^* \quad . \tag{2.33}$$

This is simply the decoupled version of (2.29). To insure equivalent misadjustment levels for the purpose of comparing transforms, the Identity transform will be used in this algorithm in place of the LMS algorithm. Power normalization also eliminates the concern for the proper choice and computation of the normalizing factor in (2.22).

An analogy helpful to the understanding of the self-orthogonalizing TRLMS filter is to envision the sliding transform of Fig 2.4 as a bank of bandpass filters [18]. For the DFT, the transform kernel (2.25) can be likened to an FIR filter $h_l(n)$ of length N with input X_n and output $s_l(n)$. For $l = 0$, $h_0(n)$ has a rectangular impulse response, giving a frequncy response of

$$H_0(e^{j\omega}) = \frac{\sin(\omega N/2)}{\sin(\omega/2)} e^{-j(n-1)\omega/2} \qquad (2.34)$$

which is a sinc-type function centered at $\omega = 0$. At the l^{th} frequency bin corresponding to the output $s_l(n)$ and center frequency $2\pi l/N$, the frequency response is simply a shifted version of $H_0(e^{j\omega})$, equal to $H_0(e^{j(\omega-2\pi l/N)})$. All of these filters will have some degree of overlap which can be lessened by increasing N or by windowing [18]. The power spectrum is whitened by appropriately scaling each of the frequency bins. For real transforms (DCT, DHT), the filter responses are generally not symmetric and may not have the same bandwidth for all frequency bins. By comparison, the KLT produces a spike at each frequency bin, implying no overlap and perfect decorrelation.

Thus, the convergence rate of the Widrow LMS algorithm can be improved at the expense of implementing an orthogonal transform and estimating the power at each of the frequency bins. Although other well understood algorithms such as RLS or the adaptive lattice may be more efficient [9], the point here is to explore the extension of the TRLMS algorithm to multiple reference inputs for application to the problem of wideband adaptive array processing. This will be discussed in subsequent chapters.

CHAPTER 3

APPLICATION OF THE TRANSFORM DOMAIN LMS ALGORITHM TO THE TWO SENSOR GSC

This chapter examines the combination of the topics discussed in the previous chapter, of the application of the TRLMS algorithm to the two sensor GSC beamformer for the purpose of improving its rate of convergence to the optimal Wiener solution. The analysis for this most simple case has essentially already been presented in Section 2.2, the only difference being the nature of the desired and reference inputs of Fig. 2.4. The main intent here is to provide a feel for the operation of the Transform Domain GSC via computer experiments.

3.1 Two Sensor Scenario

The two element (K=2) GSC shown in Fig. 3.1 was simulated on a Zenith 158 PC using FORTRAN. Sensors were spaced at a distance $d=\lambda_o/2$, where λ_o corresponds to the wavelength at the array's center frequency. A broadband signal and jammer were generated by coloring uniformly distributed zero mean pseudo-white noise with a fourth order Butterworth bandpass filter. As in Fig. 3.2, its normalized center frequency is $\omega_o = \pi/2$ with 3db bandwidth = $\pi/2$. To prevent frequency aliasing, the tap spacing of the filter W_n corresponds to half a wavelength at the maximum array frequency. All mentioned parameters were normalized to the sampling period T=1.

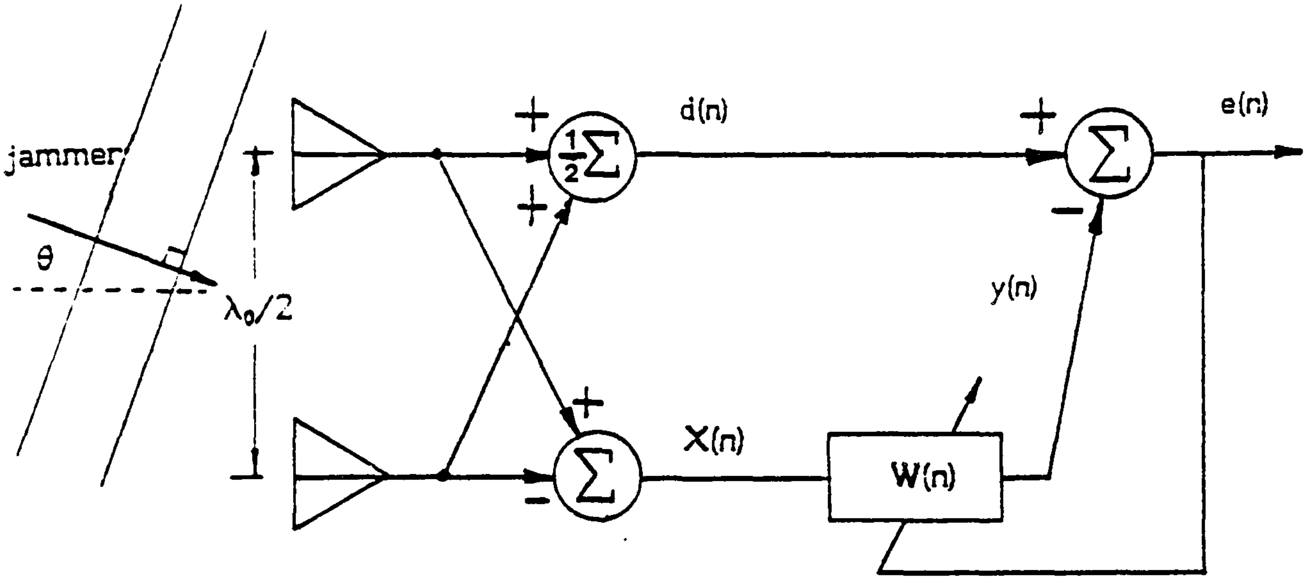


Fig. 3.1 Two sensor GSC to be simulated.

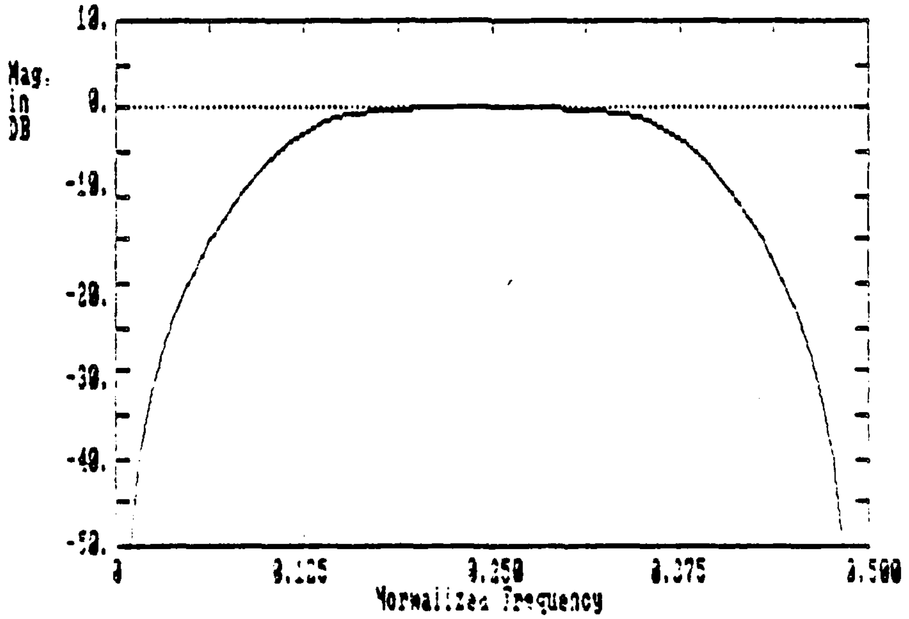


Fig. 3.2 Fourth order Butterworth filter of $\omega_0=\pi/2$, 3db BW=$\pi/2$ used to color the jammer.

To simulate a propagating wavefront incident upon the array at angle θ, from (2.2) the intersensor delay

$$\Delta = 2\sin\theta \tag{3.1}$$

was implemented using a sinc interpolator weighted by a Hamming window [12], [21]. For this class of signals, the interpolator length which gave a reasonably low mean-square error was experimentally calculated to be 11 taps.

3.2 Experimental Results

The self-orthogonalizing TRLMS algorithm of (2.33) was implemented with an Identity transform of length L=8. A jamming signal incident at 20° was set to be 20 db above the look-direction signal. The step size constant was chosen as μ =.005 with power normalization factor β =.8 and initial power estimates $\hat{\sigma}^2(0)$ = .10 . All weights were initialized to zero. The plotted output SIR curve for 15K iterations is shown in Fig. 3.3 . The mean-square error for this "learning curve" is calculated using a sliding window of 150 points.

Next examine the SIR curve for the same data sequences and parameters using the DFT, also shown in Fig. 3.3. As expected, the convergence rate is improved substantially. Even though the complex number domain of the DFT results in a complex output, as the weights converge the imaginary part becomes negligible [9], and only the real part is used in SIR calculations.

The DHT and the DCT offer improvement comparable to the DFT. All four transforms are compared in Fig. 3.4 . Note that the added complexity of the DFT is

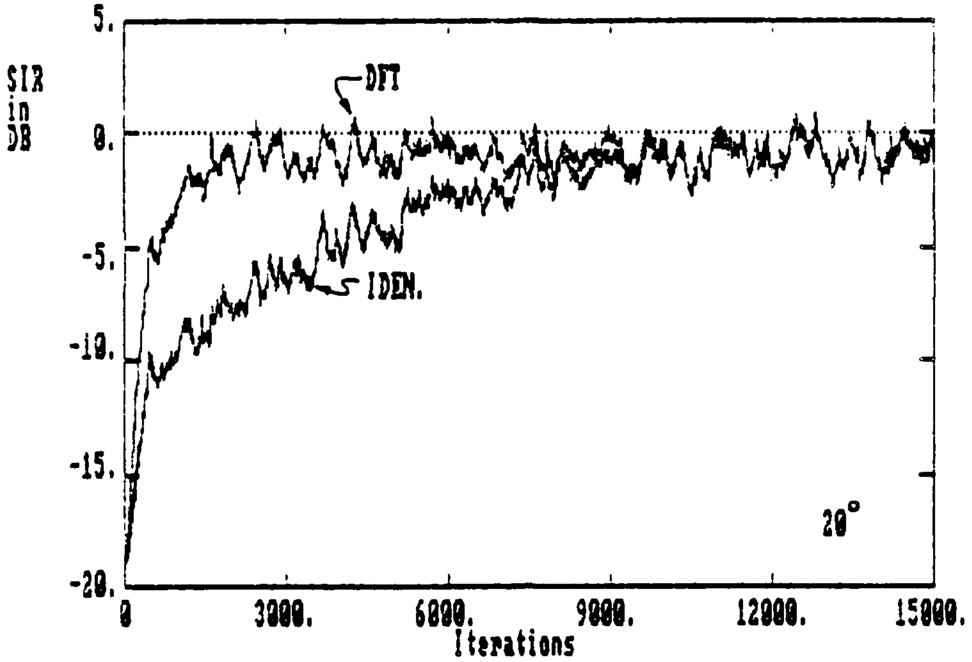


Fig. 3.3 Improvement of convergence using the DFT for this two sensor case. ($\theta=20°$, $\mu=.005$, $L=8$, $\beta=.8$)

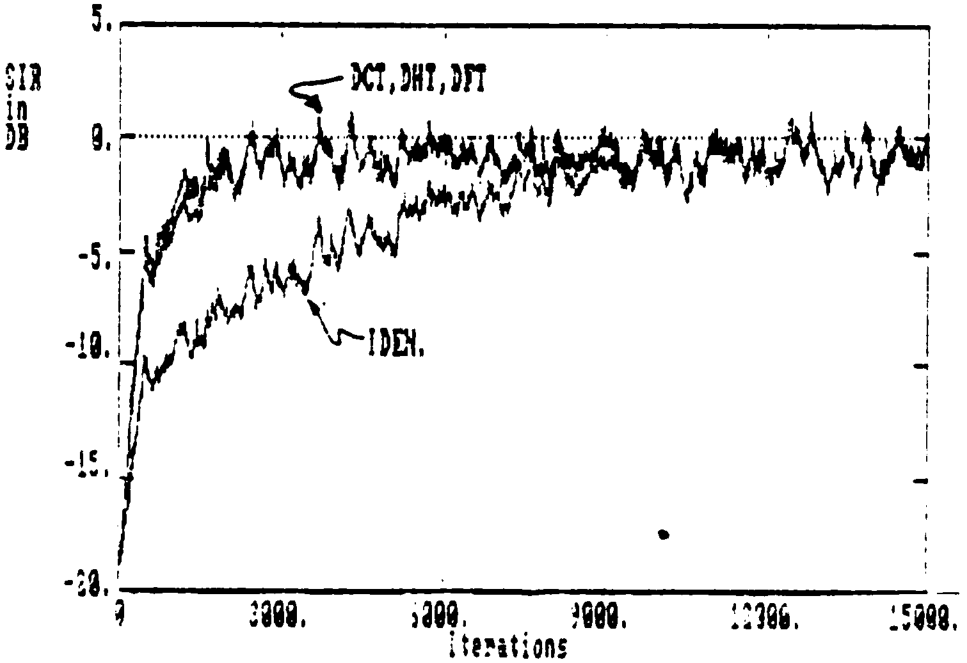


Fig. 3.4 Similarities of all transforms under consideration.

not reflected by an improvement in its performance, as the simple CHT works just as well in this case. Also, Fig. 3.4 indicates that all cases are converging to the same solution in terms of SIR.

3.3 Converged Beampatterns

Figure 3.5 shows the converged beampattern for the final weight values in terms of array gain (the ratio of power out to power in) in Db. Since the array gain is frequency dependent, the plot was generated by averaging beampatterns for discrete frequencies over the signal bandwidth [12]. The gain in the look-direction is 0 db as expected. The frequency response of the array at 20° is plotted in Fig. 3.6, demonstrating the attempt to reject the signal over its bandwidth. The gain for $\theta = 0^{\circ}$, if plotted, would show a flat response over the entire operating bandwidth. The symmetry in Fig 3.5 is due to the linear array geometry.

Notice that the initial output SIR in Figs. 3.3-3.4 does not coincide with the assigned value of -20 db. This is explained by the fact that the summation of the sensors to form the primary channel acts as a conventional beamformer, with each sensor weighted by unity. Therefore, some attenuation exists, proportional to the deviation from $\theta=0^{\circ}$. For broadband signals, this effect depends on the the spectrum of the signal. Figure 3.7 shows the initial beampattern, all weights set to zero, averaged over the signal bandwidth of $\pi/2$. As an example, the same jammer was placed at $\theta=55^{\circ}$ to coincide with the droop of the response of Fig 3.7, yielding the SIR curve of Fig. 3.8. The SIR has a head start of 6 db, but still converges to approximately the same level of 0 db. Thus it appears that for at least this case,

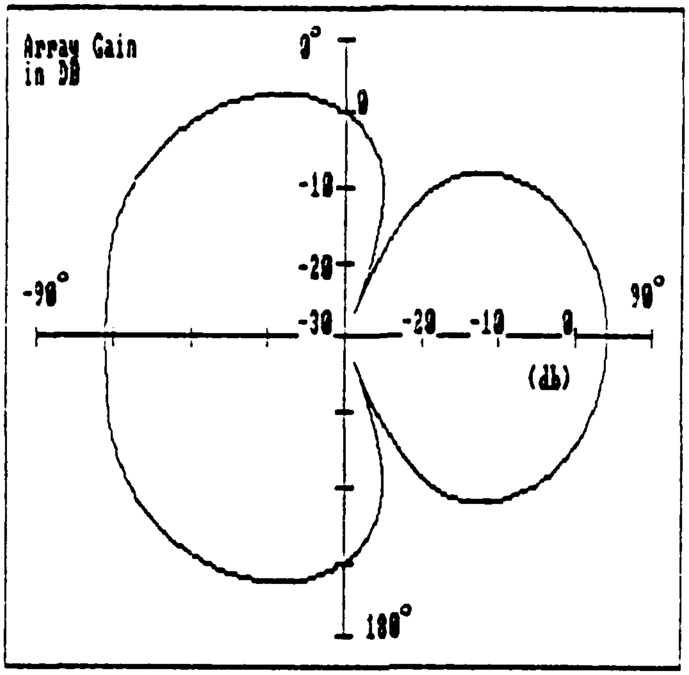


Fig. 3.5 Beampattern obtained from converged weights.

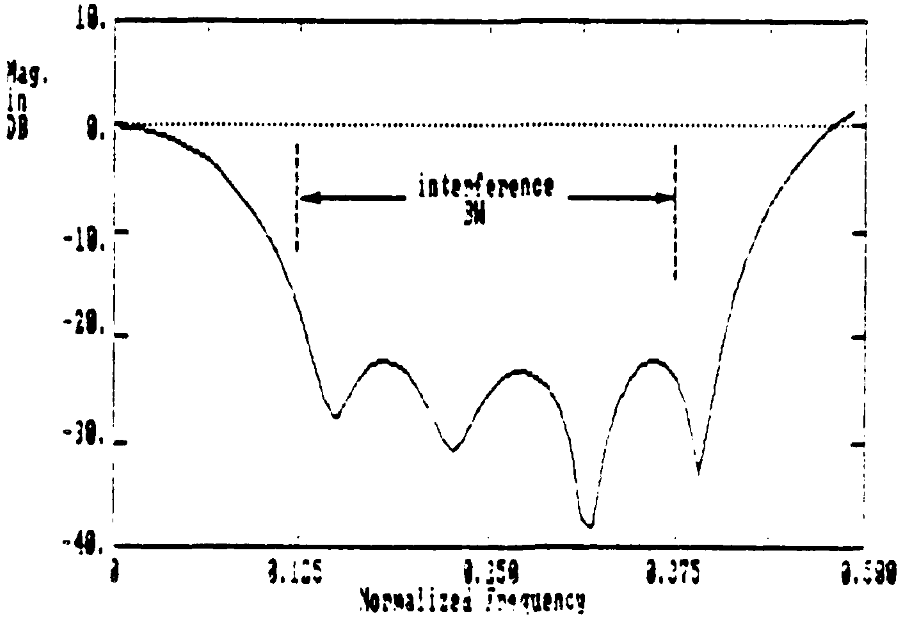


Fig. 3.6 Frequency dependent array gain at θ=20°.

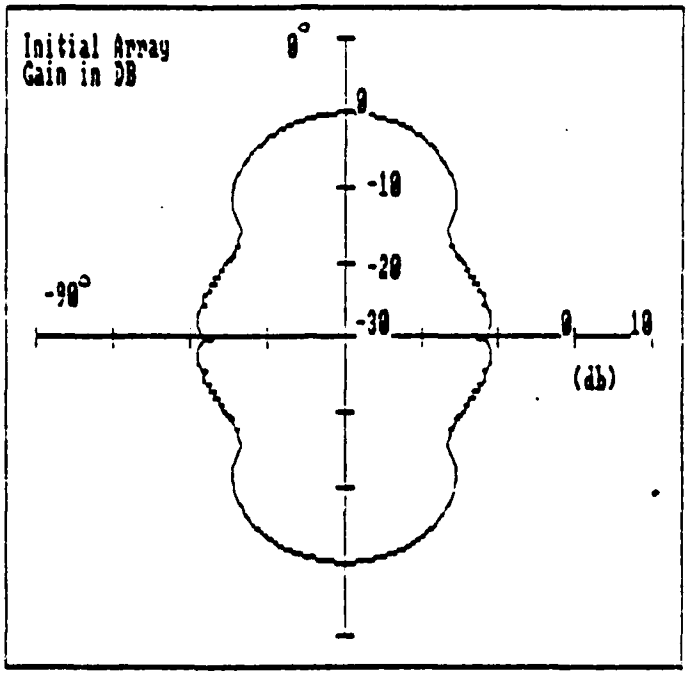


Fig. 3.7 Initial array gain (all weight values zero).

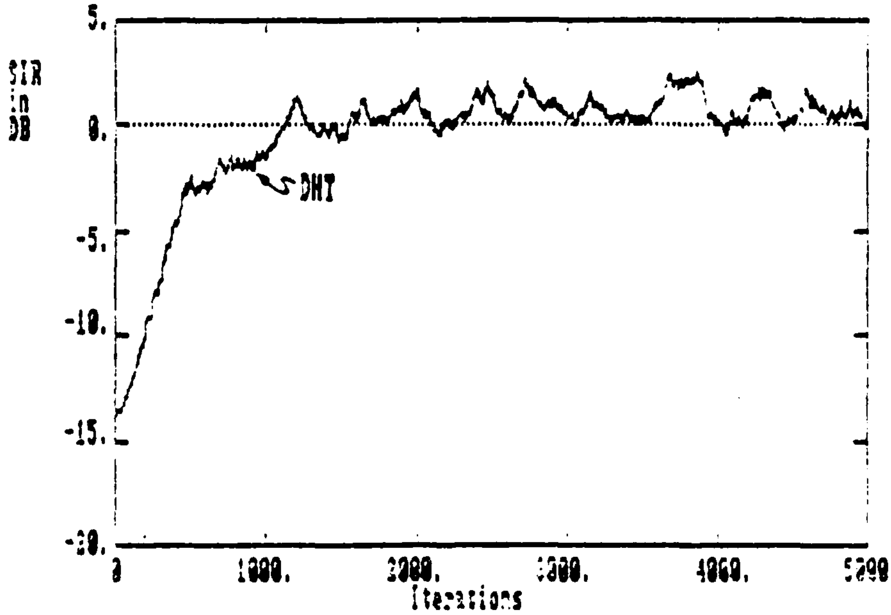


Fig. 3.8 Initial SIR offset for θ=55°.

the array has a fairly uniform rejection capability.

In general, for this simple two sensor case, the CSC appears to be very well behaved. However, for the three sensor case with two incident jammers, it will be revealed in the next chapter that this is not always the case.

CHAPTER 4

APPLICATION OF TRLMS TO THE THREE SENSOR GSC

The three sensor GSC shown in Fig. 4.1 has the capability to reject two jamming signals and pass one desired broadside signal. Convergence behavior for the one jammer case is very similar to the results presented in Chapter 3, with improved cancellation ability due to the additional degree of freedom which accompanies the extra sensor [19]. However, when a second jammer is present, the LMS algorithm can be very slow and impractical. Convergence behavior of the LMS algorithm for beamformers in general is characterized by highly disparate modes [3], [4] containing both very slow and fast components. To remedy this situation, orthogonal transforms can be used. However, for this multi-reference case, they can enter into the picture in several different ways, with efficiency dependent upon the type of transform chosen, the input data, and the ordering of the data vector X_n.

4.1 Convergence of the GSC for the LMS Algorithm

Based on experimental observations, the dependence of angle and rate of convergence will be established. Although these results are not intended to be conclusive, they serve to illustrate the motivation for the use of transforms.

To formulate the problem, two uncorrelated broadband jammers of 50% bandwidth were generated with two pseudonoise sequences and the filter with

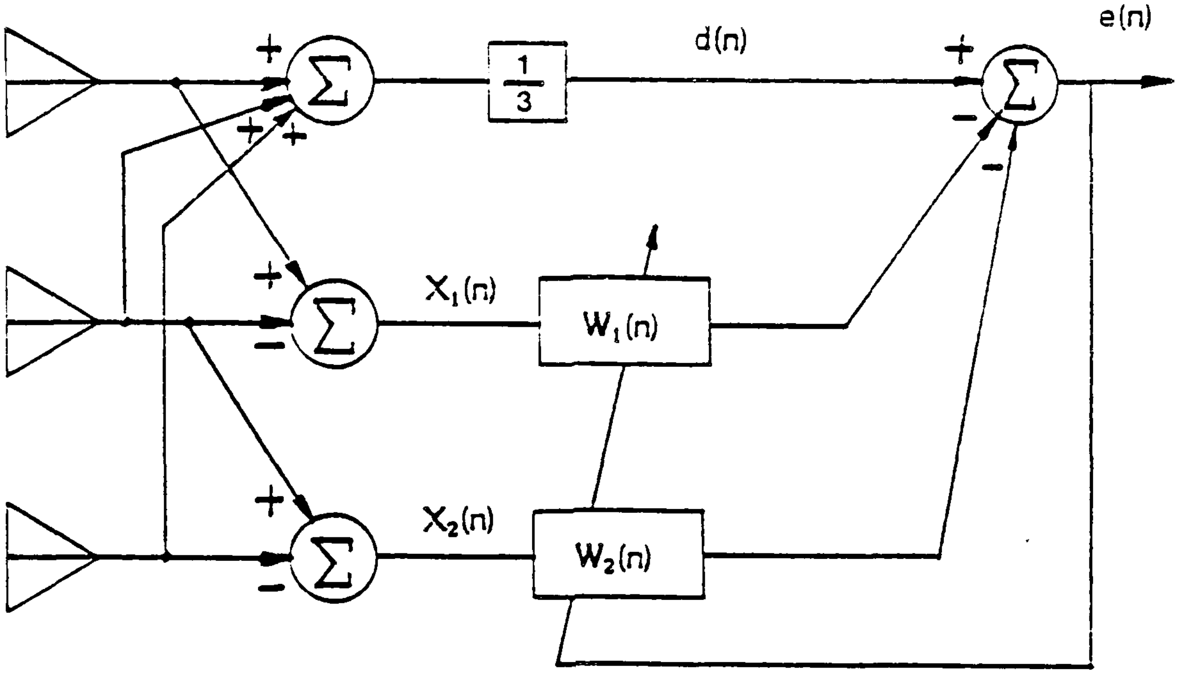
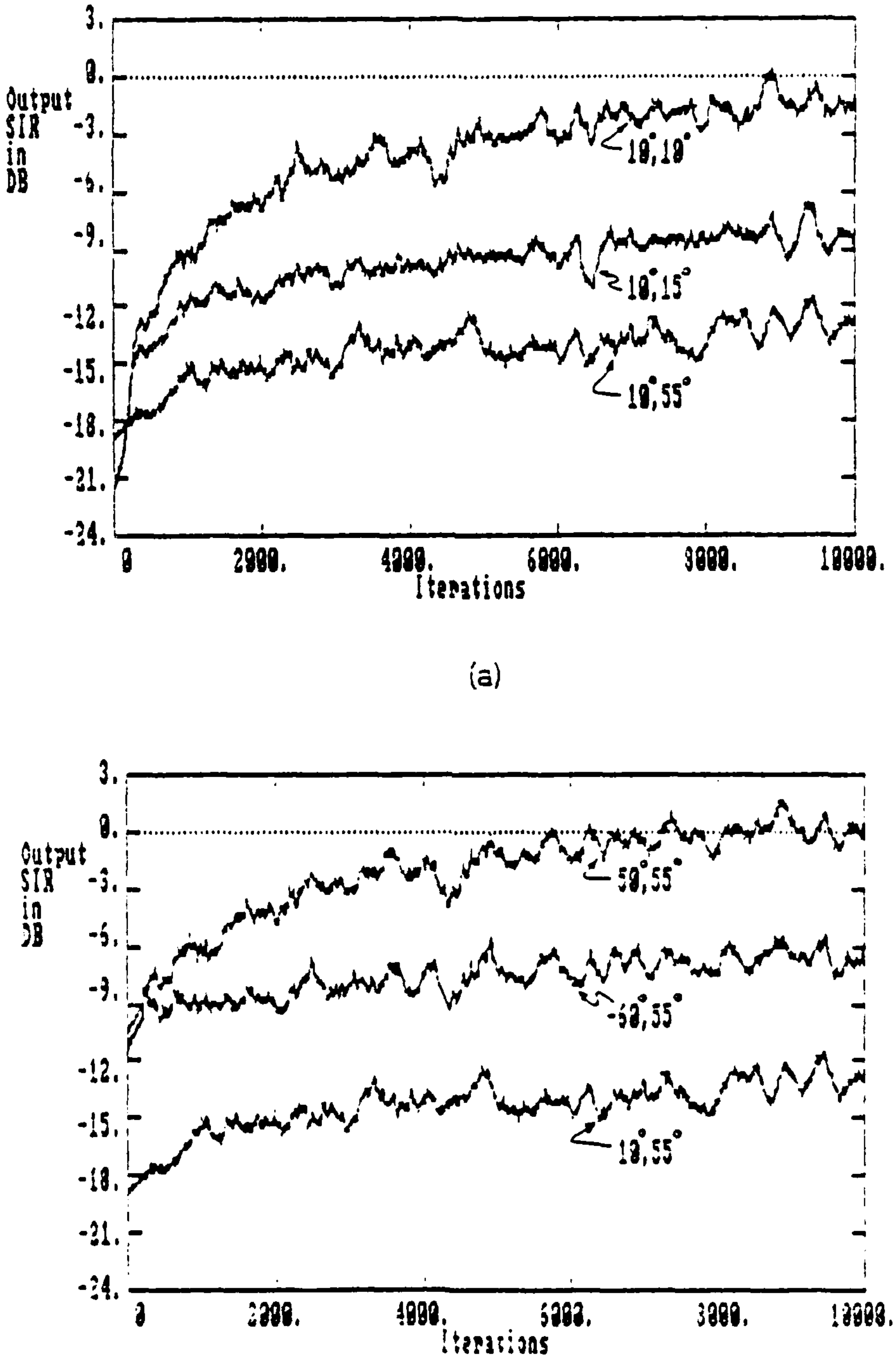


Fig. 4.1 Three sensor GSC

response plotted in Fig. 3.2. A third uncorrelated look-direction signal of -20 db was fixed at $\theta = 0^{\circ}$ as usual. All parameters and spacings are identical to those described in Section 3.2 with the exception of the filter length being L = 16 per reference channel. The self-normalizing TRLMS algorithm of (2.33) was used on each weight independently, using no transform. Jammer 1 was fixed at $\theta_1=10^{\circ}$ and Jammer 2 was placed at $\theta_2=10^{\circ}$, 15°, and 55° for three separate trials, with corresponding output SIR plots in Fig. 4.2a. The same signal sequence was used in each trial. A second set of curves for Jammer 2 fixed at $\theta_2=55^{\circ}$ and Jammer 1 placed at $\theta_1=50^{\circ}$, -60°, and (from the last set) 10° is shown in Fig. 4.2b.



(a)

(b)

Fig 4.2 Curves showing dependance of angular separation on convergence rate for (a) θ_1 fixed at 10°, and (b) θ_2 fixed at 55°.

A trend that can be observed in these curves is that the rate of convergence appears to depend upon the relative angle between the jammers. For example, the SIR levels of Fig 4.2a appear to decrease as jammer separation increases. To ensure a valid comparison between the curves of Fig 4.2, misadjustment levels must be the same [9]. Misadjustment for the LMS algorithm of (2.12) is given by

$$M \simeq \mu \, tr[R_x] \qquad (4.1)$$

The trace of R_x is known to be related to the input signal power σ_x^2 as $N\sigma_x^2$, where N is the dimension of R_x. Due to the power normalizing factor of (2.32), $\sigma_x^2 \simeq \hat{\sigma}_i^2$, and

$$M \simeq \mu N \sigma_x^2/\hat{\sigma}_i^2 \simeq \mu N \qquad (4.2)$$

Thus, the percentage of excess mean-square error [9] remains essentially constant for the various input signal powers which may arise with varying angles of incidence (see Appendix A).

Another observation reveals that the initial SIR values correspond to large deviations from the look direction. As alluded to in Section 3.3, the array elements in combination with the primary channel act as a conventional beamformer. Thus, some cancellation results from the array configuration itself, and the initial array gain for the specified 50% bandwidth is plotted in Fig. 4.3.

An important question is whether the curves of Fig 4.2 truly reflect poor convergence rates. One possibility is that the minimum MSE ξ_{min} is highly dependent upon angles of incidence, and the plots represent convergence to their

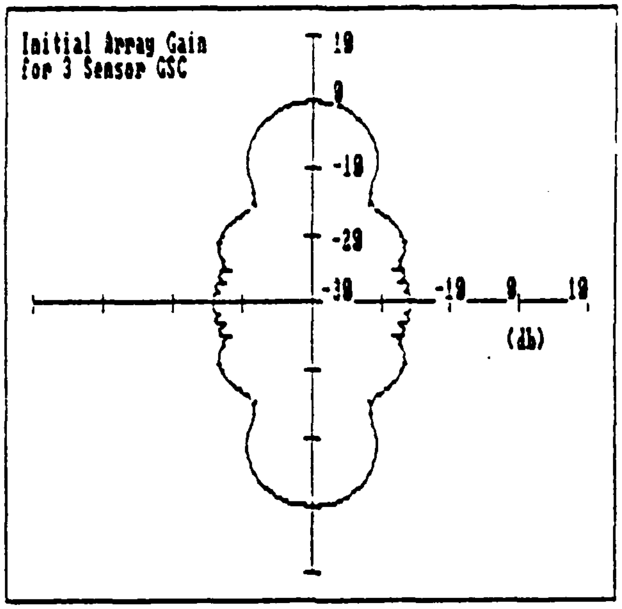


Fig. 4.3 Initial array gain for the three sensor GSC.

respective levels of ξ_{min}. It is true that ξ_{min} is angle dependent, since

$$\xi_{min} = \min\{E[e(n)^2]\} = E[d(n)^2] - P_x^T R_x^{-1} P_x \tag{4.3}$$

contains the angle dependent quantities P_x, R_x, and $E[d(n)^2]$. However, it will be shown in later sections that substantial improvement in convergence rates can be made, and variations in ξ_{min} are not as drastic as one could be led to believe. Also, by careful inspection of Fig 4.2, from the very gentle upward slopes, one can observe that the curves are still very slowly converging, even after 10^4 iterations.

For the sake of completeness, the above notion was verified by computer. A single broadband jammer was scanned across the array by one degree increments from 1° to 90°. Again a signal 20 db below the jammer was fixed at $\theta = 0°$. The weights were allowed to converge sufficiently for each angle, with results plotted in Fig. 4.4. Note that the final SIR is fairly uniform over θ. A similar result for the two sensor GSC may be found in [12]. Apart from final SIR values, similar results should be expected for the two jammer case due to linearity. This was not attempted due to the high computational expense of simulating two jammers.

This rather speculative discussion has presented some evidence suggesting that performance of the LMS algorithm can be very poor for disparate angles of arrival. Time constants of adaptation and initial SIR values may widely differ, but variations in ξ_{min} may not be as drastic as expected.

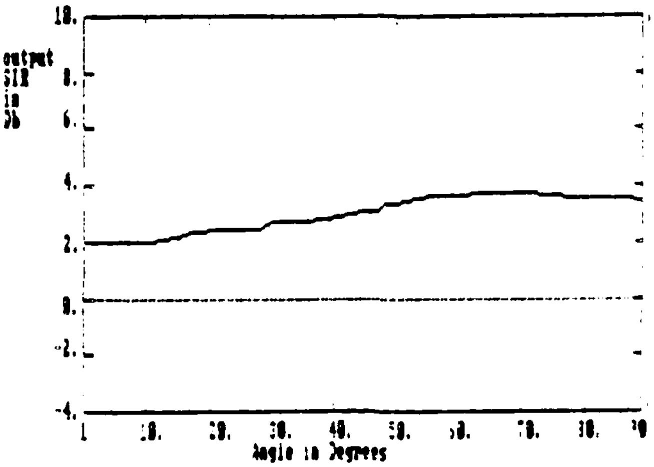


Fig. 4.4 Maximum SIR vs. angle for one jammer at -20 dB, L=16, K=3

4.2 TRLMS Applied to Individual Filters

A logical progression from Chapter 3 would be to apply a transform to X_1 and X_2 individually. Although this arrangement may provide some improvement, it does not utilize the inherent correlation between X_1 and X_2. The cross-correlation exists since both vectors contain time delayed components of the impinging signals. A derivation of power spectrum and the cross-power spectrum of the reference inputs may be found in Appendix A.

To examine the modal equations for the purpose of verifying that applying transforms in an individual fashion is suboptimal, the Wiener solution must be rederived for the individual case so the structure of the R_s matrix can be revealed. Defining the error to be

$$e(n) = d(n) - X_1^t W_1 - X_2^t W_2 \tag{4.4}$$

and taking E $[e(n)^2]$ yields

$$\begin{aligned} E[e(n)^2] = E[d(n)^2] - 2P_1^t W_1 - 2P_2^t W_2 + W_1^t R_{x11} W_1 + W_2^t R_{x22} W_2 + \\ W_1^t R_{x12} W_2 \quad W_2^t R_{x21} W_1 \quad , \end{aligned} \tag{4.5}$$

where $P_1 = E[d(n)X_1]$, $P_2 = E[d(n)X_2]$, $R_{x11} = E[X_1 X_1^t]$, $R_{x22} = E[X_2 X_2^t]$, and $R_{x12} = R_{x21}^t = E[X_1 X_2^t]$. The time index n has been dropped on the vectors and matrices for simplicity. The Wiener solution can be obtained by applying (2.10) to (4.5) directly, but the object here is to minimize (4.5) with respect to the individual vectors W_1 and W_2 to produce two separate equations, namely

$$\frac{\partial E[e(n)^2]}{\partial w_{1l}} = -2P_1 + 2W_1^t R_{x11} + 2W_2^t R_{x21} = 0 \ , \tag{4.6a}$$

$$\frac{\partial E[e(n)^2]}{\partial w_{2l}} = -2P_2 + 2W_2 R_{x22} + 2W_1^t R_{x12} = 0 \ , \tag{4.6b}$$

$$(0 \le l \le L-1) \ .$$

Equation (4.3) can be arranged in block matrix form to produce

$$W_{opt} = \begin{bmatrix} W_1 \\ W_2 \end{bmatrix}_{opt} = \begin{bmatrix} R_{x11} & R_{x12} \\ R_{x21} & R_{x22} \end{bmatrix}^{-1} \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \ . \tag{4.7}$$

The arrangment of the weight vector is identical to the stacked ordering of (2.8). Therefore, the solutions derived in Section 2.1.2 and (4.7) are identical. Equation (4.7) helps to clarify the underlying block matrix structure of the system autocorrelation matrix R_x. It contains Toeplitz autocorrelation matrices along the diagonal, and cross-correlation matrices elsewhere.

Now the effects of an orthogonal transform can be studied. Using the definitions of (2.20) and the linear properties of the transform, it can be shown by substituting (2.20) directly into (4.7) that

$$\underline{\overline{\omega}}_{opt} = \begin{bmatrix} (T^t)^{-1} W_1 \\ (T^t)^{-1} W_2 \end{bmatrix} = \begin{bmatrix} T^* R_{x11} T^t & T^* R_{x12} T^t \\ T^* R_{x21} T^t & T^* R_{x22} T^t \end{bmatrix}^{-1} \begin{bmatrix} T^* P_1 \\ T^* P_2 \end{bmatrix}, \qquad (4.8)$$

or in compact notation, $\underline{\overline{\omega}}_{opt} = R_s^{-1} P_s$. Thus, the system R_s matrix for individually applied transforms can be obtained by transforming each of the L×L submatrices of R_x as specified in (4.7).

Clearly, the R_x matrix can never be completely diagonalized when transforms are applied individually. For the optimal KLT, R_s takes on the form

$$\begin{bmatrix} \lambda_0^{11} & & & & \lambda_0^{12} & & & \mathrm{O} \\ & \lambda_1^{11} & & & & \lambda_1^{12} & & \\ & & \ddots & \mathrm{O} & & & \ddots & \\ & & & \lambda_{L-1}^{11} & & & & \lambda_{L-1}^{12} \\ \lambda_0^{12} & & \mathrm{O} & & \lambda_0^{22} & & & \\ & \lambda_1^{12} & & & & \lambda_1^{22} & & \\ & & \ddots & & & & \ddots & \\ \mathrm{O} & & & \lambda_{L-1}^{12} & & & & \lambda_{L-1}^{22} \end{bmatrix} . \qquad (4.9)$$

The eigenvalues $\lambda^{kk}{}_{i}$ along the main diagonal of (4.9) are equivalent to the power $\sigma^2{}_{ki}$ at the output of w_{ki} [17]. From (2.29), the i^{th} modal equation of subvector ω_1 becomes

$$E[\nu_{1_i}(n)] = [\, 1 - 2\mu\lambda_i^{11}/\hat{\sigma}^2{}_i]^n \nu_{1_i}(0) \;+\; [-2\mu\lambda_i^{12}/\hat{\sigma}^2{}_i]^n \nu_{i+L}(0)$$

$$\simeq [\, 1 - 2\mu]^n \nu_{1_i}(0) + [-2\mu\lambda_i^{12}/\lambda_i^{11}]^n\, \nu_{i+L}(0)\,. \qquad (4.10)$$

Thus Eq. (4.10) is clearly not decoupled due to the rightmost term. For suboptimal transforms the cross-coupling may involve additional terms.

The performance of transforms in this situation is highly dependent upon the data. For example if the matrix $R_{x11} \simeq R_{x12} \simeq R_{x22}$, then $(\lambda^{12}{}_i/\lambda^{11}{}_i) \simeq 1$ and the time constants of (4.10) would be approximately uniform over i. However, these time constants would not necessarily be minimized. From Appendix A, Equations (A.2) and (A.3) suggest that $R_{x11} \simeq R_{x12}$ for angles close to zero ($\Delta \simeq 0$). In general, some improvement is possible since R_x is at least partially diagonalized. However, this statement is only intuitive. An example will be presented in Section 4.4.1 where the use of transforms in the individual fashion actually degrades performance.

4.3 TRLMS Applied to the Stacked Vector

One way to eliminate the submatrix diagonalization of the previously discussed method is to attempt to diagonalize the *entire* R_x matrix of (4.7) by applying a transform of length 2L to the stacked data vector. The resulting structure is pictured in Fig 4.5. This proposition may at first be unsettling. Unlike the structure commonly seen in filtering applications, elements of the stacked vector X_n

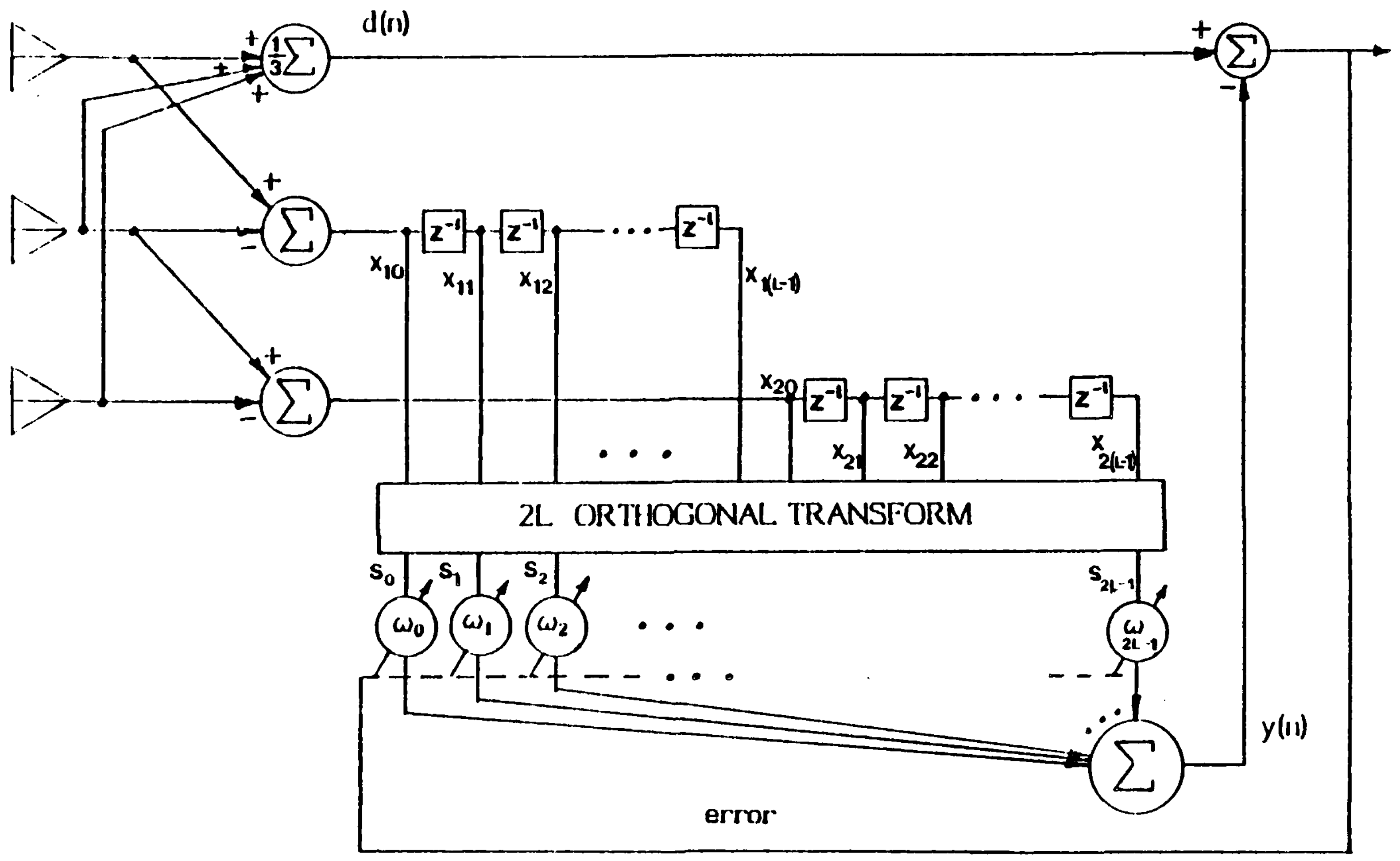


Fig. 4.5 Row-wise vector ordering for the three sensor GSC.

are no longer related by a simple unit time shift. Upon careful examination of the derivation of the TRLMS algorithm, nowhere is it required that this be the case. Each weight is treated individually, and the delay line is merely a convenient way of constructing vectors. The bandpass filter bank analogy of Section 2.2.3 is no longer as clear, but nevertheless, the input vector can still be whitened.

A requirement for the TRLMS algorithm to be effective in this case is that R_x be diagonalizable. For three or more sensors, the block matrix R_x departs from the familiar Toeplitz structure. It is, however, Hermetian symmetric (or symmetric for the real case), and a square Hermetian matrix has real eigenvalues and a unitary matrix exists which will diagonalize it [20]. The effectiveness of the transforms under consideration in this matter is yet to be revealed. Intuitively, this method should outperform the individually applied transform, due to the fact that the stacked vector transform attempts to diagonalize the system autocorrelation matrix as a whole.

4.4 Other Vector Orderings

The concept of applying the transform to the entire stacked vector X_n raises this interesting proposition: does a vector ordering exist that will make the use of a transform more effective? Thus would be a very difficult statement to prove due to the high dependence of the input parameters on the effectiveness of a transform [1]. What is ultimately desired is a general result valid over many input signal classes. Rather than search through all possible vector element orderings to find an optimal one, the intent here is to observe whether an alternate ordering makes a significant difference.

The ordering under consideration here is a "columnwise" ordering, achieved by thinking of the weights of Fig 2.3 as being arranged in rows and columns. The resulting generalized data vector has the form

$$X_n^{col} = \left\{ x_{10} \;\; x_{20} \cdots x_{K0} \cdots x_{(K-1)0} \;\; x_{11} \cdots x_{1\ell} \; x_{2\ell} \cdots x_{K\ell} \cdots x_{(K-2)(L-1)} \; x_{(K-1)(L-1)} \right\}^t . \tag{4.11}$$

For the three sensor case, the above form is constructed by simply interleaving the elements of the row vectors $X_1(n)$ and $X_2(n)$ of Eq. (2.6b). The resulting structure is shown in Fig. 4.6 . Thus, the stacked vector ordering of Fig. 4.5 can also be referred to as a "rowwise" ordering.

The column-wise ordering effectively shuffles the elements of R_x in Eq. (4.7). The elements of R_x for a row-wise ordering can be defined as

$$\left(\begin{array}{ccccc|ccccc}
\Gamma_0^{11} & \Gamma_1^{11} & \cdots & & \Gamma_{L-1}^{11} & \Gamma_0^{12} & \Gamma_1^{12} & \cdots & & \Gamma_{L-1}^{12} \\
\Gamma_1^{11} & \Gamma_0^{11} & \Gamma_1^{11} & \cdots & \Gamma_{L-2}^{11} & \Gamma_{-1}^{12} & \Gamma_0^{12} & \Gamma_1^{12} & \cdots & \Gamma_{L-2}^{12} \\
\vdots & \Gamma_1^{11} & \Gamma_0^{11} & \ddots & \vdots & \vdots & \Gamma_{-1}^{12} & \Gamma_0^{12} & \ddots & \vdots \\
\vdots & & \ddots & \ddots & \Gamma_1^{11} & \vdots & & \ddots & \ddots & \Gamma_1^{12} \\
\Gamma_{L-1}^{11} & \cdots & & \Gamma_1^{11} & \Gamma_0^{11} & \Gamma_{-L+1}^{12} & \cdots & & \Gamma_{-1}^{12} & \Gamma_0^{12} \\
\hline
\Gamma_0^{12} & \Gamma_{-1}^{12} & \cdots & & \Gamma_{-L+1}^{12} & \Gamma_0^{22} & \Gamma_1^{22} & \cdots & & \Gamma_{L-1}^{22} \\
\Gamma_1^{12} & \Gamma_0^{12} & \Gamma_{-1}^{12} & & \vdots & \Gamma_1^{22} & \Gamma_0^{22} & \Gamma_1^{22} & \cdots & \Gamma_{L-2}^{22} \\
\vdots & \Gamma_1^{12} & \Gamma_0^{12} & \ddots & \vdots & \vdots & \Gamma_1^{22} & \Gamma_0^{22} & \ddots & \vdots \\
\vdots & & \ddots & \ddots & \Gamma_{-1}^{12} & \vdots & & \ddots & \ddots & \Gamma_1^{22} \\
\Gamma_{L-1}^{12} & \cdots & & \Gamma_1^{12} & \Gamma_0^{12} & \Gamma_{L-1}^{22} & \cdots & & \Gamma_1^{22} & \Gamma_0^{22}
\end{array} \right) . \tag{4.12}$$

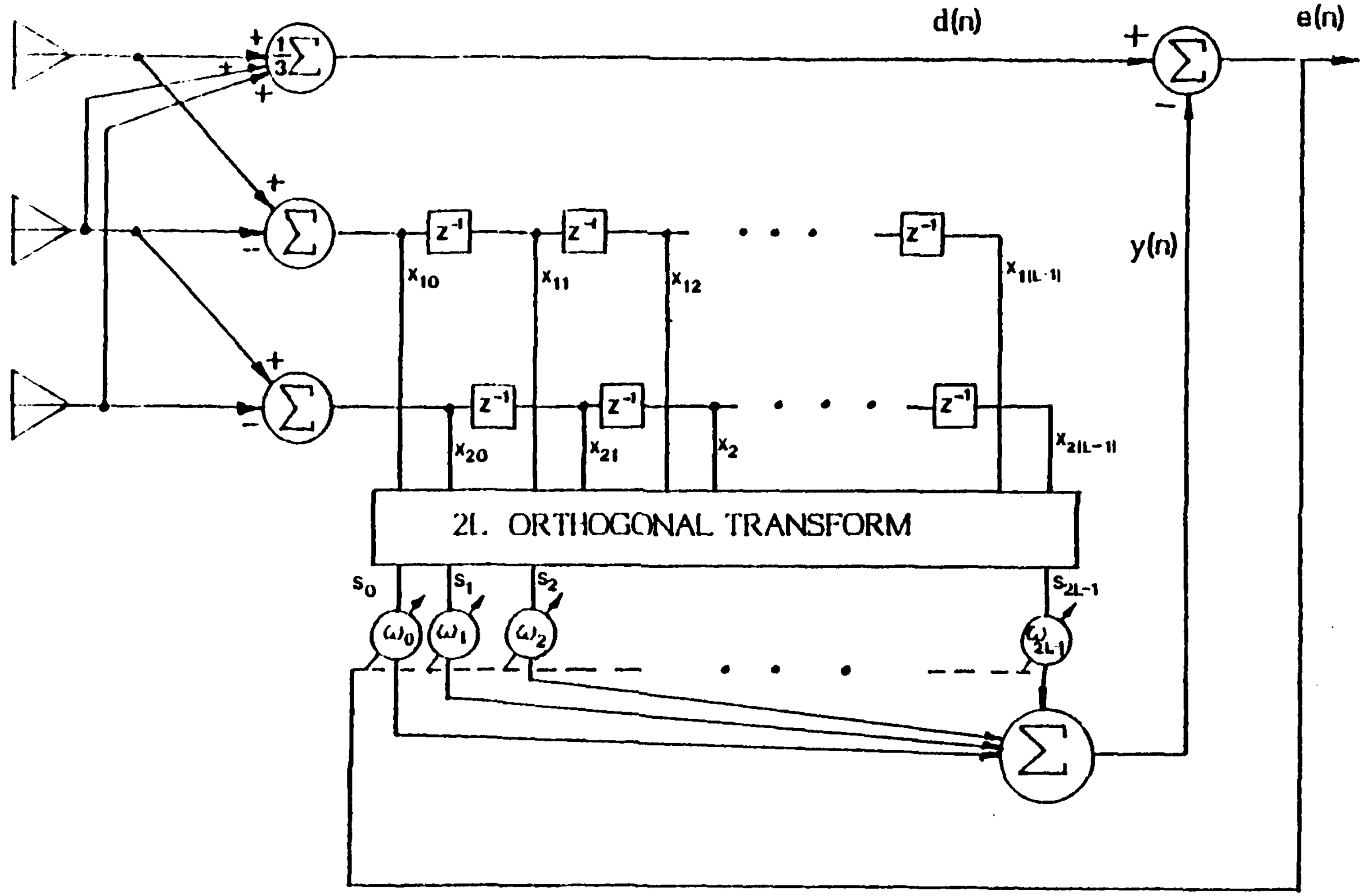


Fig. 4.6 Column-wise vector ordering for the three sensor GSC.

The submatrix R_{x12} is generally not symmetric, and negative indices are needed. Appendix A shows that $R_{x11} = R_{x22}$, but here the distinction remains to clarify the shuffling of the terms, which results in

$$\begin{bmatrix}
r_0^{11} & r_0^{12} & r_1^{11} & r_1^{12} & r_2^{11} & \cdots & r_{L-1}^{12} & r_{L-1}^{11} \\
r_0^{12} & r_0^{22} & r_{-1}^{12} & r_1^{22} & r_{-2}^{12} & \cdots & r_{L-2}^{12} & r_{L-2}^{22} \\
r_1^{11} & r_{-1}^{12} & r_0^{11} & r_0^{12} & r_1^{11} & & & \vdots \\
\vdots & \vdots & r_0^{12} & r_0^{22} & r_{-1}^{12} & \ddots & & \vdots \\
\vdots & \vdots & & \ddots & r_0^{11} & \ddots & & \\
\vdots & & & & \ddots & \ddots & & r_1^{12} \\
 & & & & & \ddots & \ddots & r_1^{22} \\
r_{L-1}^{11} & \cdots & & & & r_{-1}^{12} & r_0^{11} & r_0^{12} \\
r_{L-1}^{12} & \cdots & & & & r_1^{22} & r_0^{12} & r_0^{22}
\end{bmatrix} . \qquad (4.13)$$

The shuffling is in accordance with the arrangement of (4.11). Similar arrangements arise for four or more sensors.

Some points to be made about (4.13) are

a) If $R_{x11} = R_{x22}$ as indicated in Appendix A, then the main diagonal and every other diagonal will contain identical elements.

b) It is not Toeplitz, but still symmetric and diagonalizable.

c) It is closer to a Toeplitz structure than (4.12) since more than half of the diagonals contain equal elements.

d) The remaining diagonals have alternating elements of R_{x12}.

e) If R_{x12} is such that its higher magnitude terms are close to its main diagonal, (4.13) will have these terms closer to its main diagonal.

It is known that the DFT, DHT, and the DCT will not diagonalize a Toeplitz matrix but rather a circular matrix, a dyadic matrix, and a perturbed Toeplitz matrix, respectively [17]. Nevertheless, the performance gains of these transforms for Toeplitz structures have been well established (Chapter 3) [1], [2], [11]. Thus, the motivation behind shaping the R_x matrix into a Toeplitz-like structure should be quite clear.

For the DHT, it can be predicted that performance for the row-wise and the column-wise orderings will be identical. If the elements of the DHT matrix are shuffled in the same manner as (4.13), no change is observed. This is true for four or more sensors as well. The effects of the ordering of (4.13) on the DCT and DFT is, however, very difficult to predict, and will be heuristically examined via computer experiments in the next section.

4.5 Computer Experiments

The effects of the three vector orderings discussed will now be examined experimentally for the three sensor CSC. The same scenario as presented in Section 4.1 will be used. The examples to be presented reflect the general behavior for each case, and were selected from a large series of runs using many random number sequences. All curves on a single plot are assumed to be generated from the same underlying sequence.

As predicted, the individually applied transform performed poorly in general. Figure 4.7 shows SIR curves for the extreme cases of $\theta_1 = 10^\circ$, $\theta_2 = 10^\circ$ and

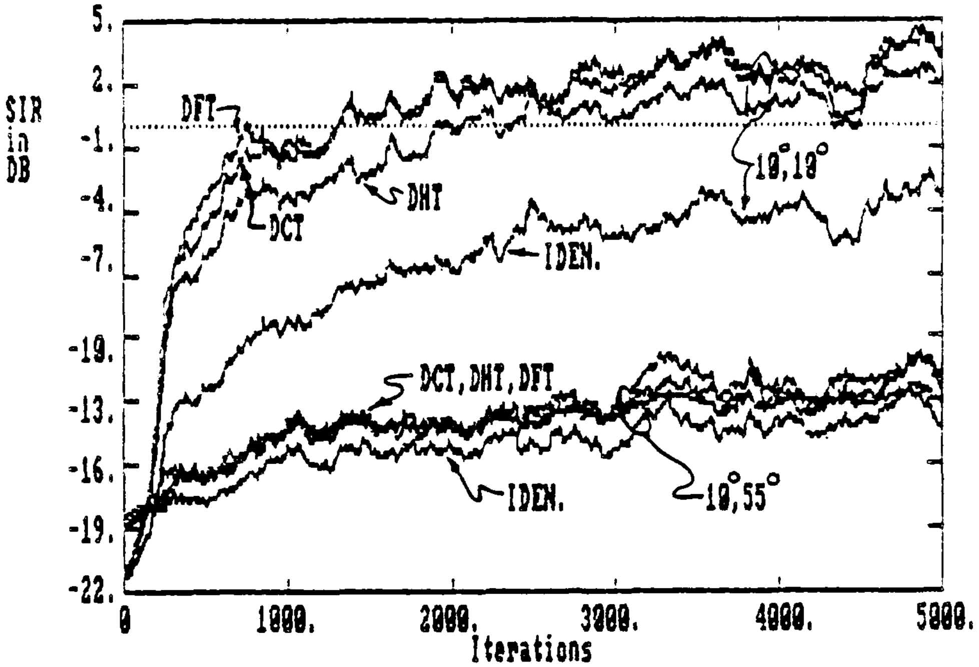


Fig. 4.7 Representative curves for performance of individually applied transforms.

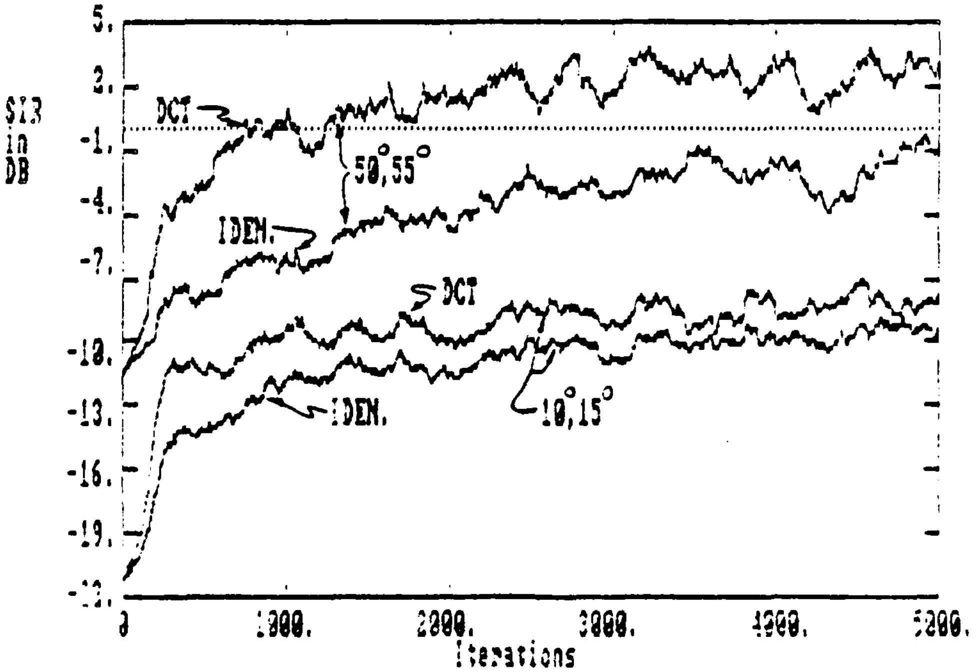


Fig. 4.8 Individually applied transforms for 5° jammer spacings, showing dependence on overall deviation from θ=0°.

$\theta_1=10^{\circ}$, $\theta_2=55^{\circ}$. Note that again the effectiveness depends upon the angular spacing. The two jammers coincident at 10° are effectively equivalent to one jammer of power ≃ 23db higher than the desired signal. Thus, improvement for one incident jammer is along the lines of that shown in Chapter 3. For closely spaced jammers as in Fig. 4.8, improvement is somewhat degraded for this 5° spacing, with effectiveness also dependent upon overall angles of incidence. Since the performance of all three transforms was very similar, only the DCT curve is shown for clarity. Significant improvement is possible not only for $\Delta \simeq 0$ as alluded to in Sec . 4.2, but for $\Delta_1 \simeq \Delta_2$ as evidenced in the upper pair of curves in Fig. 4.8. Perhaps small relative angles cause $\Phi_{12}(\omega)$ of (A.3) (and in turn R_{x12}) to be well-behaved, resulting in performance similar to the one jammer case.

Contrary to intuition, transforms applied to the row-wise ordered vector for the three sensor CSC did not produce a significant improvement over the individual case. Figure 4.9 shows a very slight improvement over the individual DCT for a pair of closely spaced and widely spaced jammers. Again, curves for the other transforms were similar and were omitted for clarity. Perhaps for this type of R_x structure, a fixed transform is ineffective in its diagonalization. Gains are still clearly visible for closely spaced jammers, however. For one jammer ($\theta_1=\theta_2=10^{\circ}$), Fig 4.10 reveals a difference among the transforms. Here the simple DHT was not as effective as the more powerful DCT. An interesting point is that the row-wise orderings of Fig. 4.10 show slightly degraded performances as compared to the individual case of $\theta_1=\theta_2=10^{\circ}$ in Fig. 4.A. Overall, for this three sensor case, the added complexity of a row-wise double length transform is not justified.

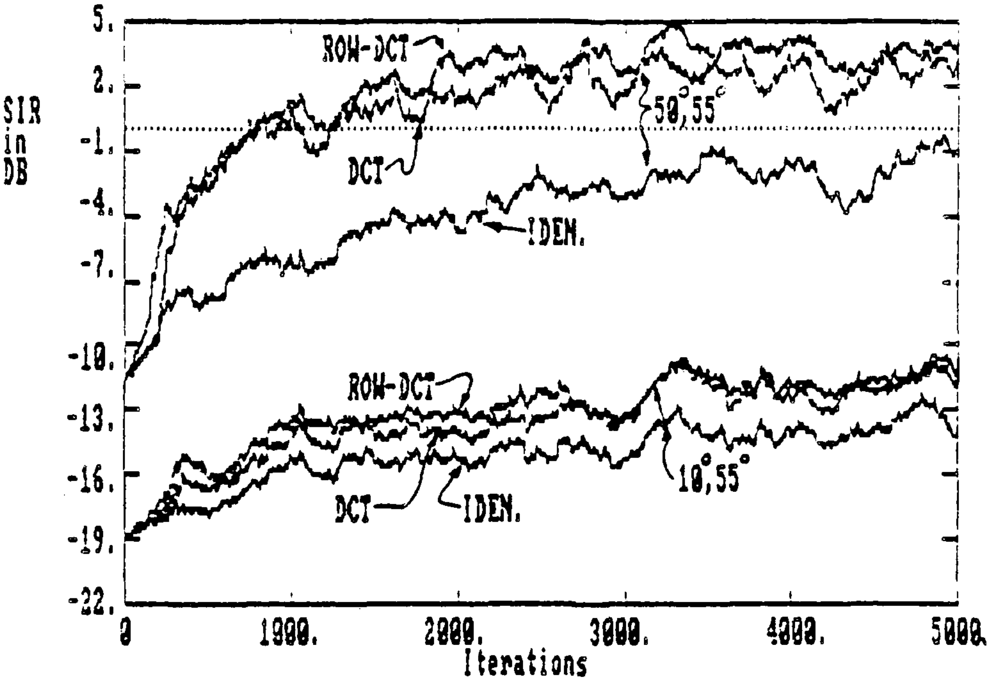


Fig. 4.9 Insignificant improvments for the rowwise ordering over the individually applied case.

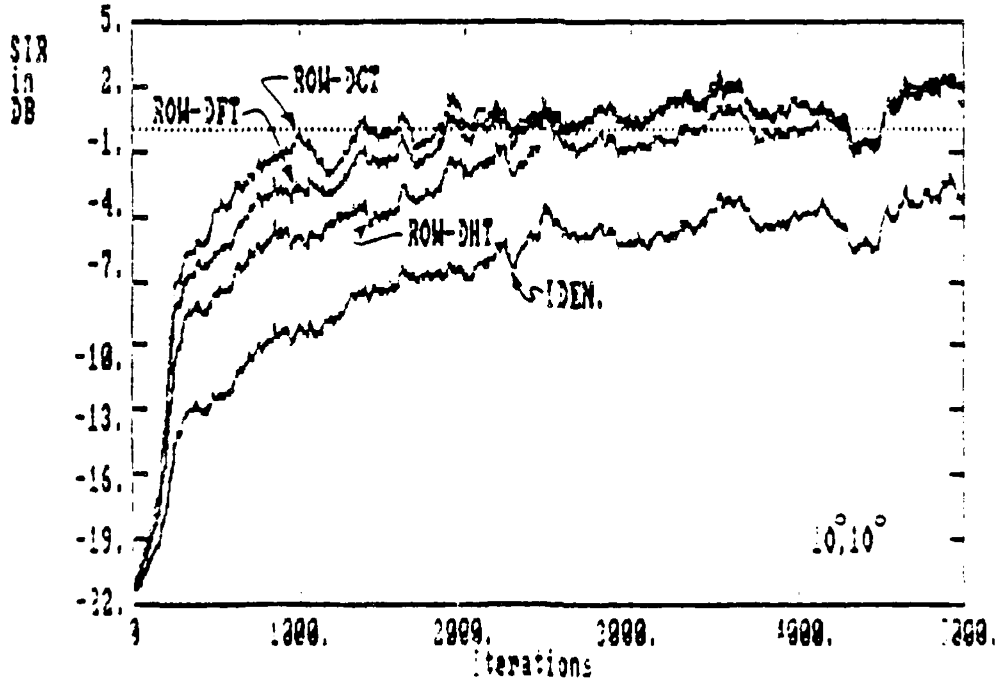


Fig. 4.10 Differences among rowwise transforms for the virtual jammer at θ=10°.

Much more promising results were discovered for the column-wise ordering, specifically for that utilizing the DCT. Figure 4.11 shows all column ordered transforms for the slowly converging θ_1=10°, θ_2=55°case. The column-DCT shows a ~6db improvement over the identity transform after 5K iterations. For this three sensor case, this was the effect generally observed for all angle spacings. To emphasize the substantial improvements possible, Fig. 4.12 shows the same curves of Fig. 4.11 for the DCT and Identity carried out for 60K iterations. Note the unreasonably slow convergence using no transform. The same was done for the θ_1=-60° and θ_2=50° case of Fig 4.2b, along with the corresponding beampattern shown in Fig. 4.13.

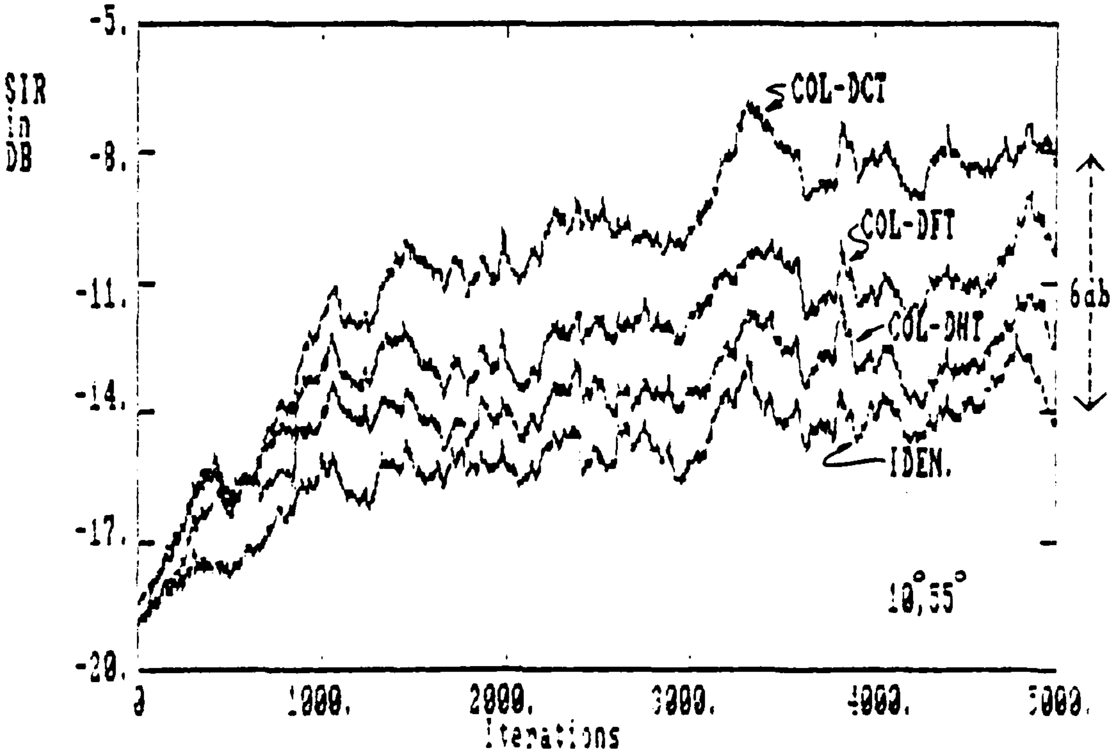


Fig. 4.11 Column-wise ordering, with the DCT showing significant improvement.

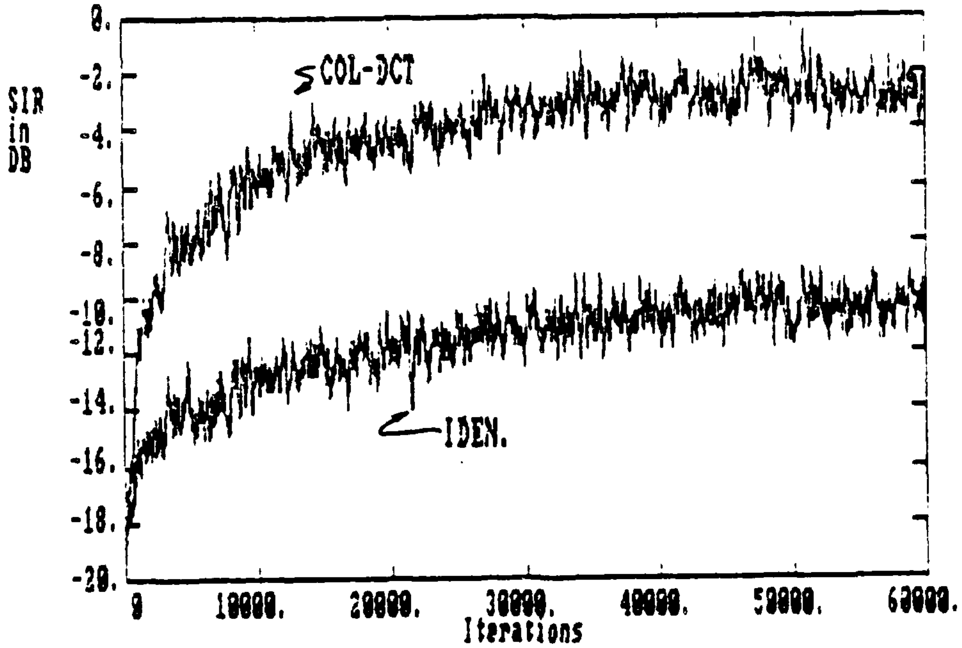
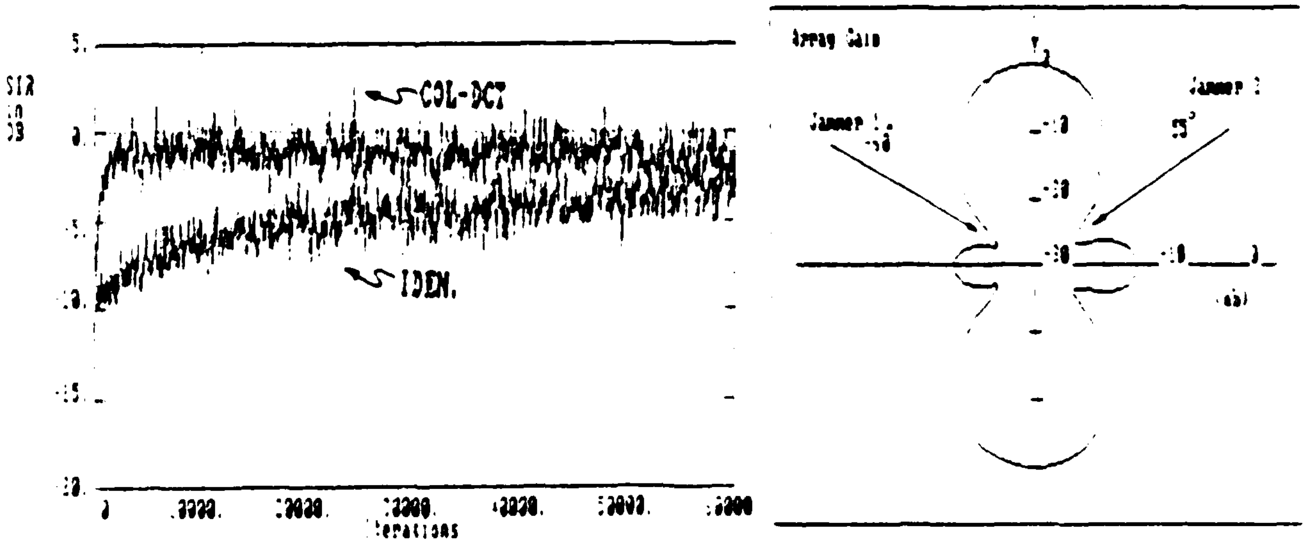


Fig. 4.12 Col-DCT curve of Fig. 4.10 carried out to 60K iterations (compared with Identity).



a)

b)

Fig. 4.13 a) Improvement for column-DCT for $\theta_1=-60^\circ$, $\theta_2=55^\circ$ and b) the corresponding beampattern.

Performance improvement for the three sensor GSC appeared to be a result specific to the column-DCT as typified by Fig. 4.14, which shows little discernibility between the row and column ordered DFT. Figure 4.15 again reveals the general trend that all transforms and orderings perform reasonably well for closely spaced jammers, as here the column-DCT is no longer outstanding.

To further study the curious effects of the column-wise DCT, the same array structure was used for jammers of different bandwidth and spectra. See Fig. 4.16. Jammer 1 was colored by a fourth order Butterworth filter of normalized center frequency $\omega_0=\pi/4$ and 3db bandwidth of $\pi/4$. Jammer 2 used a second order Butterworth filter of $\omega_0=3\pi/8$ and 3db bandwidth of $\pi/4$. The filter of Fig 3.2 was again used for the look-direction signal. Due to the longer wavelengths, longer subvector lengths were necessary, so L=32, resulting in a transform length of 64. To provide similar misadjustment levels as in previous examples, the step size was approximately halved to $\mu=.002$. For $\theta_1=-60^\circ$ and $\theta_2=55^\circ$, the output SIR is shown in Fig. 4.17 for the row and column ordered DCT. The improvement is clearly evident. Thus it appears that for at least this three sensor case for bandpass inputs, the use of a transform can be advantageous, as the column ordered DCT performs consistently well for all angles of incidence.

4.6 Examples for the Five Sensor Case

Similar experiments were done for a five sensor GSC. The main purpose here was to dismiss any misconceptions about the generality of the results obtained for the column-DCT. Jammer 1 of Fig. 4.16 was incident at $\theta_1=10^\circ$. The look-direction

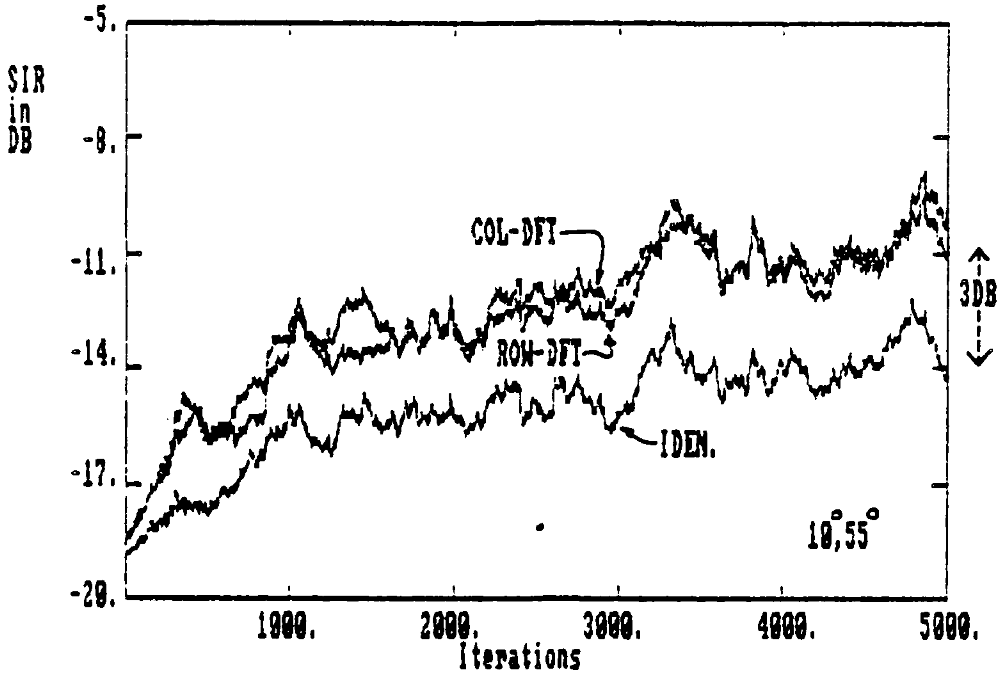


Fig 4.14 Little difference exists between DFT orderings, showing unique effectiveness of the column-DCT for the three sensor GSC.

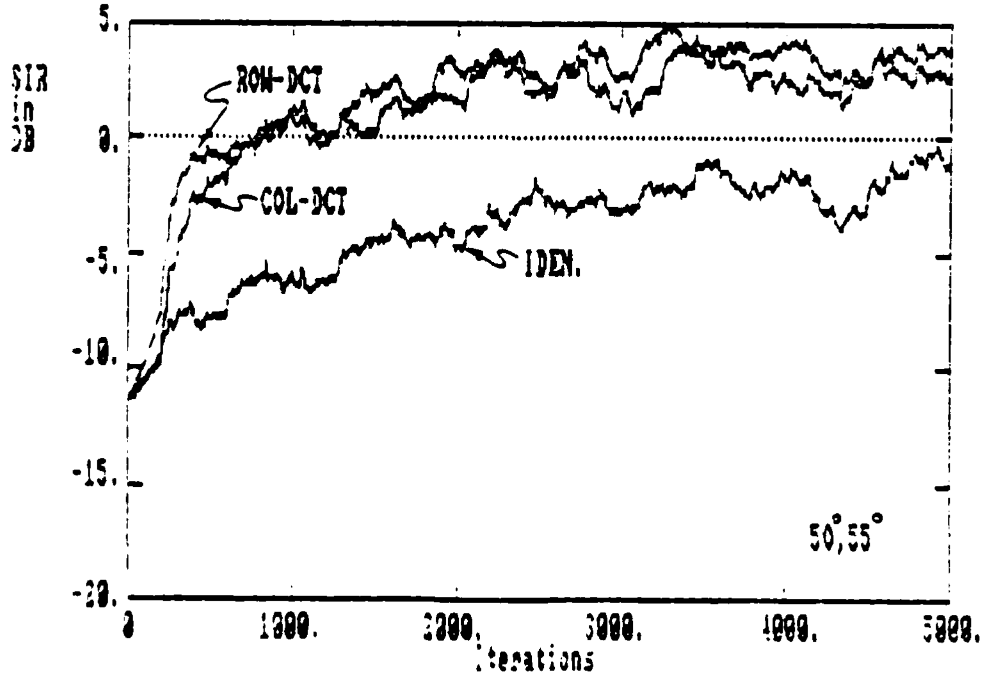


Fig. 4.15 Example showing nearly equal performance of all cases for closely spaced angles. (θ_1=50°, θ_2=55°)

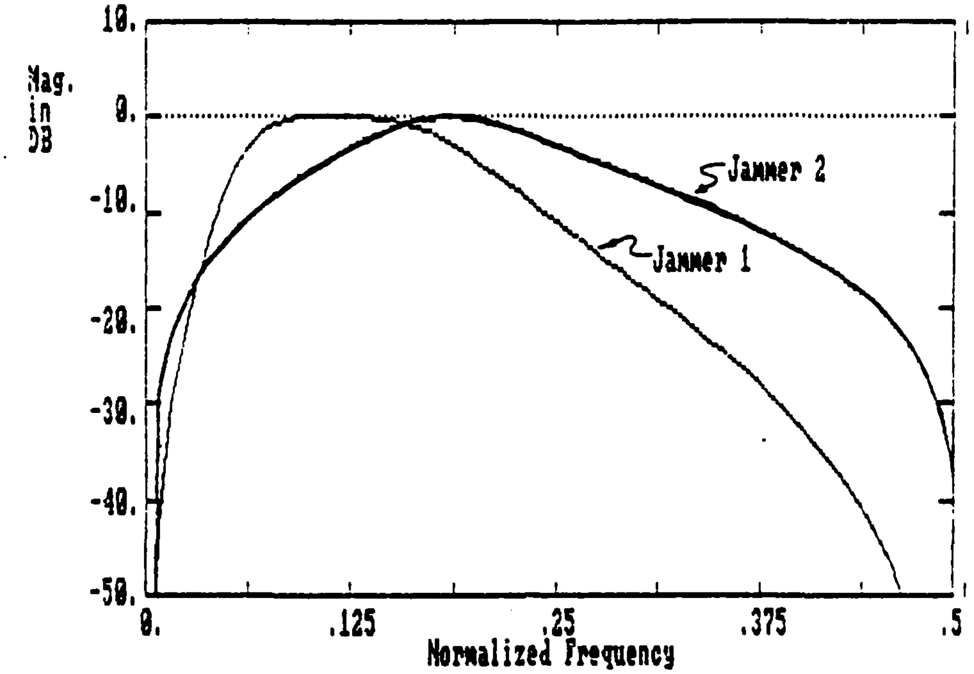


Fig. 4.16 Butterworth filters. For Jammer 1, 4th order, $\omega_0=\pi/4$, 3db BW=$\pi/4$. For Jammer 2, 2nd order, $\omega_0=3\pi/8$, 3db BW=$\pi/4$.

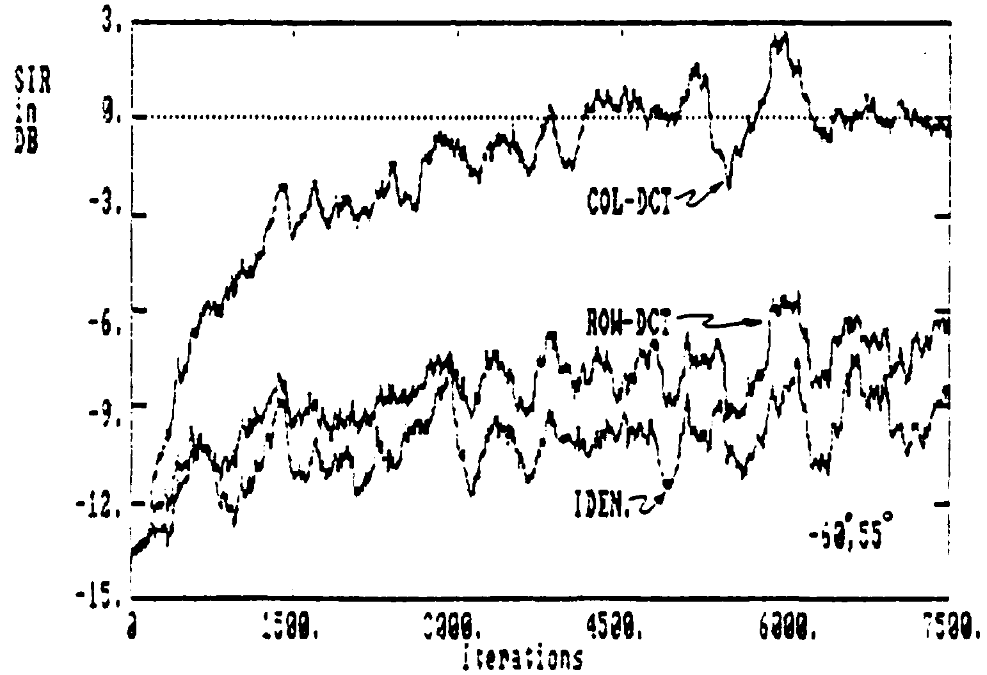


Fig. 4.17 Column-DCT for the three sensor case with inputs of Fig. 4.16, $\theta_1=-60,^{\circ}$ $\theta_2=55^{\circ}$.

signal and jammer 2 incident at $\theta_2=56^\circ$ were generated by the filter in Fig. 3.2. Results are presented in Fig. 4.18 for the indicated transforms using $\mu=.004$. and L=16, giving a transform length of 64.

The five sensor array is capable of rejecting 4 jammers, but only two were used in Fig. 4.18, leaving two extra degrees of freedom. Note that the Identity transform performed much better with five sensors than with three as pictured in say, Fig. 4.13. This is due to the extra available degrees of freedom which arise at the expense of using extra sensors [19]. This is not always of best practical interest, due to physical size or cost limitations. Sensing elements for the GSC may involve expensive analog preprocessors that require perfect balance for the blocking preprocessor of Fig. 2.2. to work effectively.

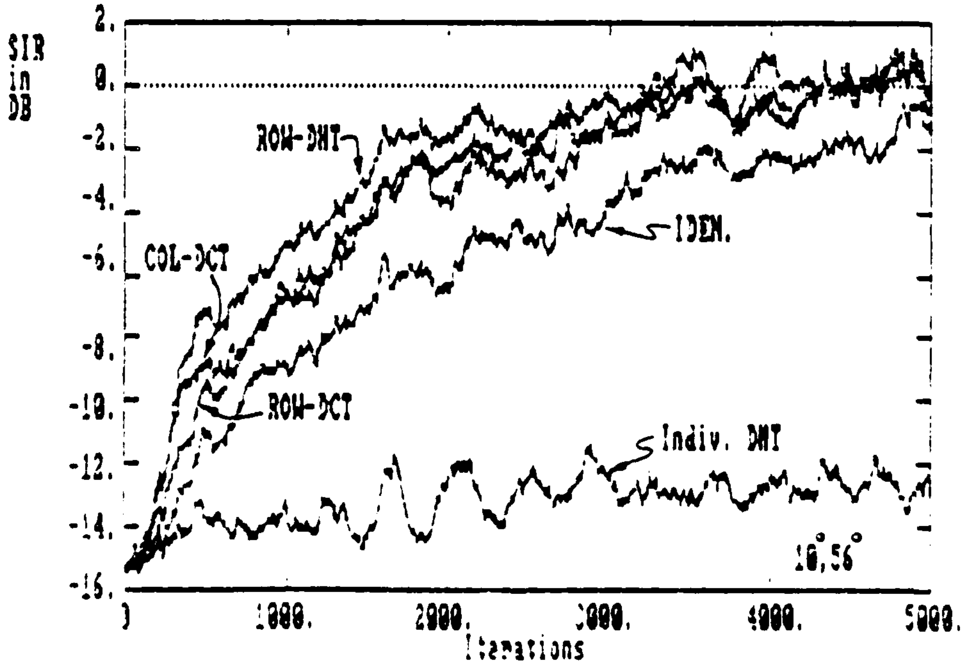


Fig. 4.18 Comparison of transforms for the five sensor GSC. $\theta_1=10^\circ$, $\theta_2=56^\circ$, $\mu=.004$. L=16.

The use of a transform is still somewhat effective according to Fig. 4.18, where ~4db improvment is achieved. Note that the simple DHT is slightly more effective in this scenario than the column-DCT, again exemplifying the high data dependence on transform efficiency. The individually applied DCT exhibits severely degraded performance, most likely due to the subdiagonalization of R_x. Also, the distinctive improvements shown by the column-DCT in the previous section are not seen here.

A third jammer was placed at θ_3=-30° to remove one of the degrees of freedom. This resulted in a generally slower convergence rate, as evident from the 25K samples needed in Figs. 4.19. Again, up to ~4 db improvement was achieved, and no noticeable difference is seen between the row and column ordered DCT shown in Figs. 4.19a and 4.19b, respectively. Unlike Fig. 4.18, the DHT of Fig. 4.19c did not do as well here.

In general, convergence of these five sensor cases is fairly good without a transform, so improvements with a transform are not expected to be great. This is possible at the expense of extra sensing elements. Clearly, performance is also dependent upon the number of sensors, and generalizations cannot be made from the three sensor case alone.

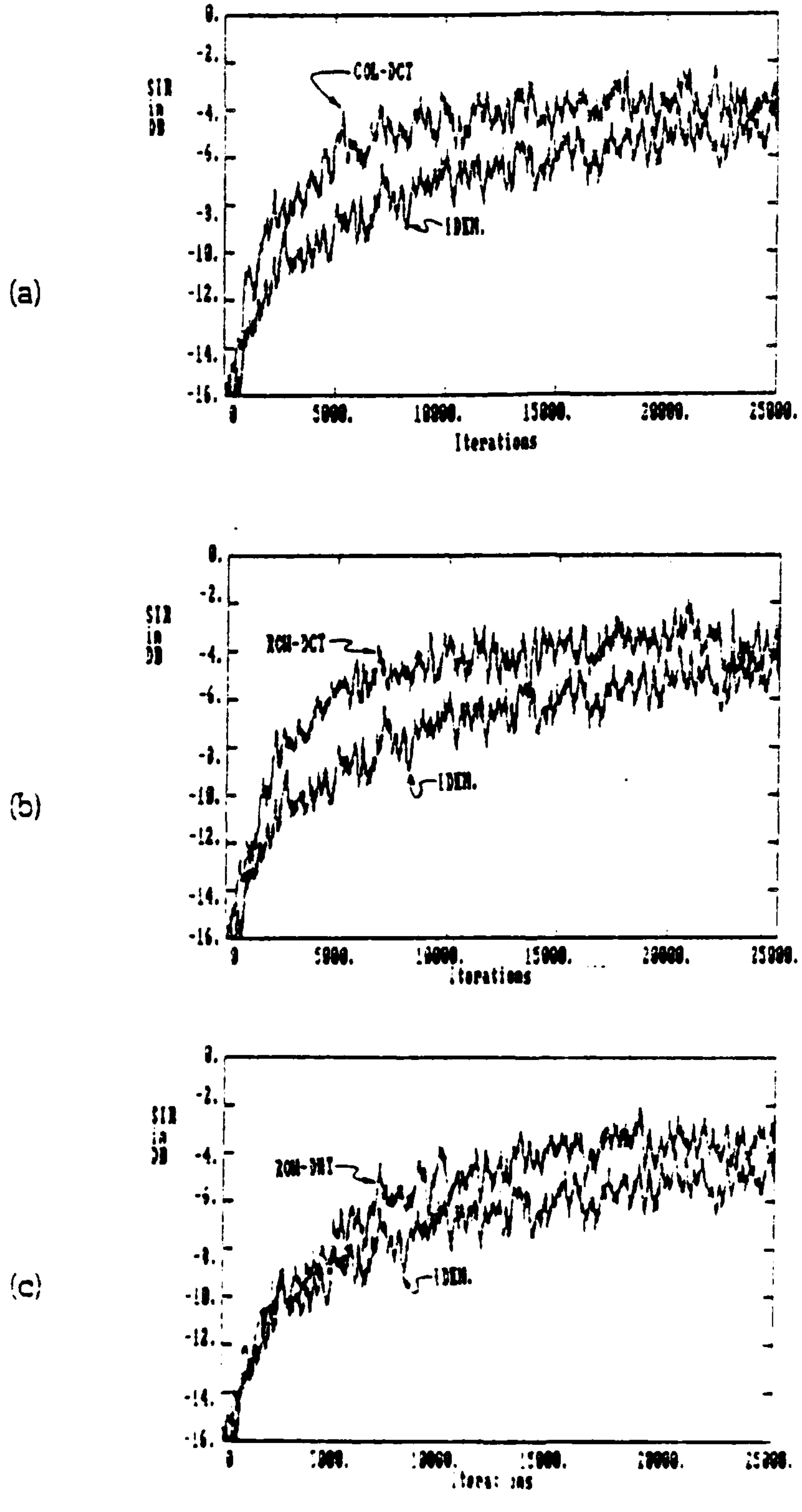


Fig 4.13 Five sensor case with three jammers (θ_1=10°, θ_2=66°, θ_3=-30°). Comparison of identity to (a) column-DCT, (b) row-DCT, (c) row-DHT.

CHAPTER 5

CONCLUSIONS

After a brief review of the Generalized Sidelobe Canceller and the Transform Domain LMS algorithm, the two were combined successfully to improve the convergence rate for the two sensor case. The performance was very well-behaved for all three transforms under consideration (DFT, DHT, DCT), due to the Toeplitz nature of the autocorrelation matrix R_x.

According to an analogy of the GSC to multiple reference noise cancelling, it was shown that the results of extending the TRLMS to two (or more) references are not as definitive. Many arrangements of the data vector X_n are possible. Speculation on the performance of a transform for three vector orderings was given: the individual case, the row-wise ordering, and the column-wise ordering.

The three sensor case was the main example. For the class of wideband inputs considered, performance of individually applied transforms fared well for closely spaced jammers only. For other angles, poor performance was attributed to the subdiagonalization of R_x.

In an attempt to diagonalize the entire R_x matrix, double length transforms were used. Of all transforms and orderings for the three sensor case, the column-wise DCT appeared to be uniquely effective. It performed consistently well over all angles and many different bandwidths. Perhaps the DCT can diagonalize the type of R_x structure which arises from this ordering, which was shown to be near-Toeplitz.

Interestingly, Jayant and Noll [17] state that the DCT can diagonalize slightly perturbed Toeplitz matrices. The DCT is well known to provide near optimal performance in numerous other applications, including image coding and channel equalization [16], [17], so its distinction here should not come as a surprise. All other combinations showed poor performance, except for the cases of closely spaced jammers, where most all combinations gave good results.

The five sensor case was examined briefly for two and three jammers. It was observed that the convergence rate decreased as jammers were added. Possible improvements were not as drastic, as distinctions between row and column orderings were slight. The main purpose here was to show that the results for the columnwise DCT are not general. However, some improvement was still possible. It was also revealed here that individually applied transforms can actually degrade performance substantially over the transform-less case, again an effect of subdiagonalization.

Overall, it was found that improvement of convergence rates is possible using fixed orthogonal transforms, but performance is highly dependent upon input data and the array processing structure. The desired result of being able to have one transform that works well for all signals and numbers of sensors seems unlikely. In an adaptive environment, very little is usually known about incoming signals. Therefore, more research is needed to develop criteria for choosing a transform based on simple known parameters, such as the number of sensors, the ordering of the data vectors, or perhaps some figure of merit which can be calculated from a few data samples.

It appears that the key lies in the understanding of the array and data dependent structures of R_x. Many researchers have used the Hilbert-Schmidt (weak) norm [16] to calculate the residual correlation of a transformed matrix. This figure of merit measures the ability of a transform to decorrelate its inputs. This has yet to be successfully related to the modal time constants of the TRLMS algorithm. This was attempted in the course of this research but not presented, since no meaningful trends could be observed. A more suitable measure may be necessary.

Perhaps other transforms not considered here, such as the Haar or Discrete Sine Transforms [16] may be more effective. Analysis and simulation of the optimal KLT should be performed. This may reveal any trends and similarites between the optimal basis functions and those of a fixed transform under consideration. Also, other vector orderings may exist (possibly optimal) which will shape R_x into a structure more compatible with certain transforms.

Also not attempted here is the comparison of TRLMS to the performance and computational efficiency of other algorithms. Perhaps other techniques exist which are more predictable. For example, a similar analysis for the multichannel adaptive lattice as applied to the CSC can be found in [7].

Also, a more careful study of the scenarios presented would be informative. Examining effects of thermal noise and other array imperfections would help assess its practicality. Simulations using other classes of data inputs and correlated inputs, such as may arise from multipath interference, should be done to test the validity of the generalizations made from the experiments presented here. Interesting results may be observed if the five sensor array with four incident jammers were simulated. This would use all available degrees of freedom as the two and three sensor simulations did.

Finally, the extension of the TRLMS algorithm to multiple inputs may also be generalized to other applications. The various ways a transform can be applied were brought to attention.

In conclusion, the analysis and experimental results presented here were intended to examine the possibility of using fixed orthogonal transforms in an adaptive beamforming scenario. Behaviors for some simple inputs were categorized and some promising results were obtained. This study suggests that further analysis and evaluation of the TRLMS beamforming algorithm may lead to more useful results in the future.

REFERENCES

[1] W. K. Jenkins and D.F. Marshall, "On the analysis and design of unitary transformations for adaptive FIR digital filters," *Proceedings of the 29th Midwest Symposium on Circuits and Systems*, Lincoln, NE, August 11-12, 1986, to appear.

[2] J. C. Lee and C. K. Un, "Performance of transform-domain LMS adaptive digital filters," *IEEE Trans. Acoust., Speech,. Signal Processing*, vol. ASSP-34, pp. 499-510, June 1986.

[3] R. Compton Jr., "Improved feedback loop for adaptive arrays," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-16, no. 2, March 1980.

[4] M. Klemes, "A practical method of obtaining constant convergence rates in LMS adaptive arrays," *IEEE Trans. on Antennas Propag.*, vol. AP-34, March 1986.

[5] K. Gerlach, "Fast orthogonalization networks," *IEEE Trans. on Antennas. Propagat.*, vol. AP-34, March 1986.

[6] L. J. Griffiths, "Adaptive structures for multiple input noise cancelling applications," *ICASSP '79 Proc.*, pp. 925-928, Washington, D.C.

[7] B. Chang and D. H. Youn, "Multichannel lattice filter for an adaptive array processor with linear constraints," *ICASSP '86 Proc.*, vol. 3, pp. 1829-1832, 1986.

[8] W. D. White, "Cascade preprocessors for adaptive antennas," *IEEE Trans. Antennas Propag.*, vol AP-24, no. 5, pp. 670-684, September 1976.

[9] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1985.

[10] O. L. Frost, "An algorithm for linearly constrained adaptive array processing," Proc. *IEEE*, vol. 7, no. 7, pp. 926-935, August 1972.

[11] L. J. Griffiths and C. W. Jim, "An alternative approach to linearly constrainted adaptive beamforming," *IEEE Trans. Antennas Propag.*, vol. AP-30, pp. 27-34, January 1982.

[12] R. P. Gooch and J. J. Shynk, "Wide-band adaptive array processing using pole-zero digital filters," *IEEE Trans. Antennas Propag.*, vol. AP-34, no. 3, March 1986.

[13] B. Widrow *et al.*, "Adaptive noise cancelling: principles and applications," *Proc. IEEE*, vol. 63, no. 12, pp. 1692-1716, December 1975.

[14] L. Ljung and T. Soderstrom, *Theory and Practice of Recursive Identification*. Cambridge, MA: MIT Press, 1983.

[15] A. Gersho, "Adaptive equalization of highly dispersive channels for data transmission," *Bell Syst. Tech. J.*, vol. 48, pp. 55-70, January 1969.

[16] K. R. Rao, *Discrete Transforms and Their Applications*. Van Nostrand Reinhold: New York, NY., 1985.

[17] N. S. Jayant and P. Noll, *Digital Coding of Waveforms, Principles and Applications to Speech and Video*. Prentice Hall: Englewood Cliffs, NJ., 1984.

[18] L. R. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1975.

[19] J. T. Mayhan, A. J. Simmons, and W. C. Cummings, "Wide-band adaptive antenna nulling using tapped delay lines," *IEEE Trans. Antennas Propag.*, vol. AP-29, no. 6, November 1981.

[20] A. Giordano and F. Hsu, *Least Squares Estimation with Applications to Digital Signal Processing*. New York: Wiley, 1985.

[21] B. C. Mather, "Fourier Domain Interpolation Techniques for Synthetic Aperture Radar," Technical Report UILU-ENG-86-2226, Coordinated Science Laboratory, University of Illinois, Urbana, 1986.

APPENDIX A

DERIVATION OF POWER SPECTRA FOR THE THREE SENSOR CSC

A derivation of power spectrum and the cross-power spectrum of the reference inputs may be useful in the understanding of the correlations between them. Let $\Phi_1(\omega)$ and $\Phi_2(\omega)$ be the spectral densities of jammer 1 and jammer 2 respectively, where ω is normalized digital frequency. Without loss of generality, let the topmost sensor of Fig 4.1 be used as a reference where the respective delays $\Delta_1 = \Delta_2 = 0$. The transfer function for the i^{th} jammer to the k^{th} reference input (through the preprocessor) is

$$H_i^k(z) = Z^{-(k-1)\Delta_i} - Z^{-k\Delta_i} \quad . \qquad (A.1)$$

The power spectral density at the k_{th} reference is

$$\Phi_{x_{kk}}(\omega) = |H_1^k(\omega)|^2\Phi_1(\omega) + |H_2^k(\omega)|^2\Phi_2(\omega) \qquad (A.2a)$$

$$= 2\Phi_1(\omega)\ (1 - \cos\omega\Delta_1) + 2\Phi_2(\omega)\ (1 - \cos\omega\Delta_2) \quad . \qquad (A.2b)$$

The cross-power spectrum between reference channels can be calculated from $\Phi_{x12}(\omega)=\Phi_{x11}(\omega)H_{12}(\omega)$, where $H_{12}(\omega)$ is the transfer function between the first and second reference inputs. In the Z domain $H_{12}(z) = z^{-\Delta_i}$. This yields

$$\Phi_{x_{12}}(\omega) = \left\{ |H_1^1(z)|^2 \Phi_1(z) z^{-\Delta_1} + |H_2^1(z)|^2 \Phi_2(z) z^{-\Delta_2} \right\} \Big|_{z=e^{j\omega}}$$

$$=2e^{-j\omega\Delta_1}\,\Phi_1(\omega)\,(1-\cos\omega\Delta_1)+2e^{-j\omega\Delta_2}\,\Phi_2(\omega)\,(1-\cos\omega\Delta_2)\,, \qquad (A.3b)$$

and also $\Phi_{x_{21}}(\omega) = \Phi_{x_{12}}(\omega)^*$. Note that when $\Delta_i = 0$ in (A.3b), $\Phi_{x_{kk}}(\omega) = 0$ in agreement with the blocking behavior of the preprocessor. Also the terms in (A.3b) are related to the those in (A.2b) by a complex exponential, corresponding to the intersensor delay Δ_i. The autocorrelation functions can be obtained from the above equations by taking a simple inverse DTFT.

APPENDIX B

FORTRAN CODE LISTING FOR THE THREE SENSOR GSC

```
      PROGRAM JAS2D8
C
C ****************************************************
C    SIMULATES A THREE CHNL GRIFFITHS-JIM BEAMFORMER
C    WHICH USES REAL TRANSFORMS FOR DECORRELATION.
C ****************************************************
C
      REAL*8 ARGS(10),BREG(10),CREG(10),LOOKDIR(300),JAMMER1(300),
     +  JAMMER2(300),J1A(300),J1B(300),J2A(300),J2B(300),ZB(0:64),
     +  DELAY(4),MYU,MYUN(300),ZOLD1(64),ZA(0:64),NOISEC(300),
     +  W(0:90),ZSAVE(0:300),E(300),PWRE,MSE,PI,Y(300),BETA,
     +  D(300),ZMAX,ZD(300),NUMREP,NOISEA(1000),NOISEB(300),
     +  SINC(0:64),WIN(0:64),Z1(0:64),NERR,ZOLD2(64),SIGREP,
     +  Z2(0:64),Z3(0:64),Z4(0:64),S(0:256),SIGMA(0:128,0:1),
     +  J(0:128),JE(0:128),SIR
      INTEGER*4 BLKSIZE,ANGLE1,ANGLE2,TALLY,ITER,LENGTH,P,DSEED
      INTEGER ADJ(2),LEN,X1(2),X2(2),DIGIT
C
C  SET PARAMETERS
      PRINT*,'  ENTER SEED NUMBER'
      READ* DSEED
      PRINT*,'  ENTER # OF ITERATIONS IN MULTIPLE OF 250'
      READ* ITER
      PRINT*,'  ENTER CONVERGENCE FACTOR MU'
      READ*,MYU
      PRINT*,'  ENTER ORDER OF TRANSFORM'
      READ* N
      PRINT*,'  CHOOSE TRANSFORM'
      PRINT*,'     0 - DCT OF DOUBLE LENGTH'
      PRINT*,'     1 - DST OF DOUBLE LENGTH'
      PRINT*,'     2 - NO TRANSFORM, NORMLSE'
      PRINT*,'     3 - DCT of alternating samples'
      PRINT*,'     4 - DST OF ALTERNATING SAMPLES'
      READ* DIGIT
C  SIZE OF INTERPOLATOR=2M+1
      M=9
      BLKSIZE=250
      PI=3.14159265
      NUMREP=4
      LENGTH=BLKSIZE+2*M+2
      NERR=100
      PWRE=0
      TALLY=0
      BETA=.6

      PRINT*,'  ENTER DESIRED INITIAL SIR'
      READ* SIR
      SIR=10.**(SIR/20.)
C FIRST PASS FOR PROPER INITIALIZATION
      LEN=3*LENGTH
      CALL RANDOM(NOISEA,LEN,DSEED)
C MAKE ZERO MEAN NOISE (VARIANCE = [illegible])
C JAMMER1 [illegible] DB [illegible] LARGER
C JAMMER2 [illegible] DB [illegible] LARGER
      DO [illegible] I=1,LENGTH
        NOISEA(I)=SIR*(NOISEA(I)-.5)
        NOISEB(I)=(NOISEA(I+LENGTH)-.5)
        NOISEC(I)=(NOISEA(I+2*LENGTH)-.5)
[illegible]    CONTINUE
C  INITIALIZE FILTER REGISTERS
        DO [illegible] I=1,[illegible]
          AREG(I)=0.
          BREG(I)=0.
          CREG(I)=0.
[illegible]   CONTINUE
C
C  GENERATE SAMPLE [illegible] NOISE SIGNALS
        CALL [illegible] NOISEA LOOKDIR [illegible] BLKSIZE
        CALL [illegible] [illegible] SIR JAMMER1 [illegible] LENGTH
        CALL [illegible] [illegible] SIR JAMMER2 [illegible] LENGTH
C
C ****************************************************
C   [illegible]
C ****************************************************
        [illegible]
C       [illegible]                                    DEGREES
        READ* ANGLE1
        PRINT 811
  811   FORMAT (' ENTER DIR. OF JAMMER2 (BW=PI/2) IN DEGREES',/)
        READ*,ANGLE2
        PRINT 812
  812   FORMAT (' TYPE 0 TO REMOVE DESIRED SIGNAL, 1 TO KEEP: ')
        READ*,KD
        DELAY(1)=-2.*DSIN(REAL(ANGLE1)*PI/180.)
        DELAY(2)=-2.*DSIN(REAL(ANGLE2)*PI/180.)
C
C  CHOOSE CORRECT DELAYS FOR INTERPOLATOR
        DO 45 I=1,2
          IF (DELAY(I).LT.-1.5) THEN
            DELAY(I)=DELAY(I)+1
            DELAY(I+2)=2*DELAY(I)+1
            ADJ(I)=3
            X1(I)=2
            X2(I)=0
          ELSEIF (DELAY(I).LT.-1.) THEN
            DELAY(I)=DELAY(I)+1
            DELAY(I+2)=2*DELAY(I)
            ADJ(I)=2
            X1(I)=1
            X2(I)=0
          ELSEIF (DELAY(I).LT.-.5) THEN
            DELAY(I+2)=2*DELAY(I)+1
            ADJ(I)=1
            X1(I)=1
            X2(I)=0
          ELSEIF (DELAY(I).LE..5) THEN
            DELAY(I+2)=2*DELAY(I)
            ADJ(I)=0
            X1(I)=0
            X2(I)=0
          ELSEIF (DELAY(I).LE.[illegible]) THEN
            DELAY(I+2)=2*DELAY(I)-1
            ADJ(I)=0
            X1(I)=0
            X2(I)=1
          ELSEIF (DELAY(I).LE.1.5) THEN

            DELAY(I)=DELAY(I)-1
            DELAY(I+2)=2*DELAY(I)
            ADJ(I)=-1
            X1(I)=0
            X2(I)=1
          ELSE
            DELAY(I)=DELAY(I)-1
            DELAY(I+2)=2*DELAY(I)-1
            ADJ(I)=-1
            X1(I)=0
            X2(I)=2
          ENDIF
 45     CONTINUE
        DO 70 K=1,4
C HAMMING WINDOW
          DO 100 I=0,2*M
            WIN(I)=.54-.46*DCOS(2*PI*(REAL(I)+DELAY(K))/
     +           REAL(2*M+2))
 100      CONTINUE
C interpolating functions
          DO 120 I=-M,M
            IF (DELAY(K).NE.0) THEN
              SINC(I+M)=DSIN(PI*(DELAY(K)+REAL(I)))/
     +               (PI*(DELAY(K)+REAL(I)))
            ELSE
              SINC(I+M)=1.
            ENDIF
            [illegible]
 120      CONTINUE
 70     CONTINUE
C  [illegible] functions [illegible] different [illegible]
        DO [illegible]
          [illegible]
          [illegible]
          [illegible]
          [illegible]
        CONTINUE
```

```
C
C  INITIALIZE BEAMFORMER
      DO 150 I=0,2*L-1
         XA(I)=0.
         XB(I)=0.
         W(I)=0.
         SIGMA(I,J)=.5
150   CONTINUE
      OPEN(8,FILE='D:SIR.DAT')
      WRITE(8,900) ITER
C
C ****************************************************************
C     MAIN LOOP
C ****************************************************************
      DO 199 TALLY=0,ITER-1,BLKSIZE
         PRINT*,TALLY,MSE
         CALL INTERP(JAMMER1,J1B,Z1,LENGTH,M)
         CALL INTERP(JAMMER1,J1A,Z3,LENGTH,M)
         CALL INTERP(JAMMER2,J2B,Z2,LENGTH,M)
         CALL INTERP(JAMMER2,J2A,Z4,LENGTH,M)

C Arrange Interpolator Outputs for Proper Delay
         DO 190 I=1,BLKSIZE
            JAMMER1(I)=JAMMER1(I+M+ADJ(1))
            JAMMER2(I)=JAMMER2(I+M+ADJ(2))
            J1A(I)=J1A(I+K2(1))
            J1B(I)=J1B(I+K1(1))
            J2A(I)=J2A(I+K2(2))
            J2B(I)=J2B(I+K1(2))
190      CONTINUE
C
C      * TRANSFORM DOMAIN LMS ALGORITHM *
         DO 200 K=1,BLKSIZE
C  shift registers
            DO 210 I=L-1,1,-1
               XA(I)=XA(I-1)
               XB(I)=XB(I-1)
210         CONTINUE
            D(K)=KD*LOOKDIR(K)+(JAMMER1(K)+JAMMER2(K)+J1B(K)+
     &           J1A(K)+J2A(K)+J2B(K))/3.
            XA(0)=J1A(K)+J2A(K)-J1B(K)-J2B(K)
            XB(0)=J1B(K)+J2B(K)-JAMMER1(K)-JAMMER2(K)
            Y(K)=0.
C        N X N ORTHOGONAL TRANSFORM
            IF (DIGIT.GT.2) THEN
C  Column-wise ordering
               DO 211 I=0,L-1
                  D(2*I)=XA(I)
                  D(2*I+1)=XB(I)
211            CONTINUE
               IF (DIGIT.EQ.3) THEN
                  CALL DCT(D,TX,2*L)
               ELSEIF (DIGIT.EQ.4) THEN
                  CALL FFT(D,TX,2*L)
               ENDIF
            ELSE
C Row-wise ordering
               DO 213 I=0,L-1
                  D(I+L)=XB(I)
                  D(I)=XA(I)
213            CONTINUE
               IF (DIGIT.EQ.3) THEN
                  CALL DCT(D,TX,2*L)
               ELSEIF (DIGIT.EQ.4) THEN
                  CALL FFT(D,TX,2*L)
               ELSE
                  DO 214 I=0,2*L-1
                     TX(I)=D(I)
214               CONTINUE
               ENDIF
            ENDIF
C       STEP SIZE NORMALIZATION (ORTHOGONALIZATION)
            DO 215 I=0,2*L-1
               SIGMA(I,J)=BETA*SIGMA(I,J)+(1-BETA)*TX(I)**2
215         CONTINUE
C        PERFORM FILTERING
            DO 220 I=0,2*L-1
               Y(K)=Y(K)+TX(I)*W(I)
220         CONTINUE

C    OUTPUT OF ARRAY IS Y(K)
            E(K)=D(K)-Y(K)
            ER(K)=E(K)-KD*LOOKDIR(K)
C    UPDATE COEFFICIENTS
            DO 230 I=0,2*L-1
               W(I)=W(I)+2*MYU*E(K)*TX(I)/SIGMA(I,J)
230         CONTINUE
200      CONTINUE
C
C    LEARNING CURVE STUFF  ASSUMED TO BE ERGODIC
C
C   Generate learning curve reference points.
         IF (TALLY.EQ.0) THEN
            SIGREF=0.
            DO 240 I=0,199
               ESAVE(I)=(D(I+1)-KD*LOOKDIR(I+1))**2
               SIGREF=SIGREF+LOOKDIR(I+1)**2
               PMSE=PMSE+ESAVE(I)
240         CONTINUE
            SIRREF=PMSE
            DO 195 I=200,BLKSIZE-1
               SIRREF=SIRREF+(D(I+1)-KD*LOOKDIR(I+1))**2
               SIGREF=SIGREF+LOOKDIR(I+1)**2
195         CONTINUE
            SIRREF=10.*DLOG10(SIGREF/SIRREF)
            MSEREF=PMSE/NERR
         ENDIF
C
C  Sliding window of 200 points
         DO 250 I=1,BLKSIZE
            P=MOD(I+TALLY-1,200)
            PMSE=PMSE-ESAVE(P)
            ESAVE(P)=ER(I)**2
            PMSE=PMSE+ESAVE(P)
            MSE=PMSE/NERR
            MSE=10.*DLOG10(MSEREF/MSE)+SIRREF
            WRITE(8,901) MSE
250      CONTINUE
C
C  GENERATE NEXT BLOCK OF DATA FOR 250 MORE ITERATIONS
         IF (TALLY.LT.ITER-BLKSIZE) THEN
            DO 300 I=1,2*M+2
               HOLD1(I)=JAMMER1(I+BLKSIZE)
               HOLD2(I)=JAMMER2(I+BLKSIZE)
300         CONTINUE
            CALL RANDOM(NOISEA,3*BLKSIZE,DSEED)
            DO 305 I=1,BLKSIZE
               NOISEA(I)=SIR*(NOISEA(I)-.5)
               NOISEB(I)=(NOISEA(I+BLKSIZE)-.5)
               NOISEC(I)=(NOISEA(I+2*BLKSIZE)-.5)
305         CONTINUE
            CALL FILTER(NOISEA,LOOKDIR,AREG,BLKSIZE)
            CALL FILTER(NOISEB,JAMMER1,BREG,BLKSIZE)
            CALL FILTER(NOISEC,JAMMER2,CREG,BLKSIZE)
            DO 310 I=BLKSIZE,1,-1
               JAMMER1(I+2*M+2)=JAMMER1(I)
               JAMMER2(I+2*M+2)=JAMMER2(I)
310         CONTINUE

            DO 320 I=1,2*M+2
               JAMMER1(I)=HOLD1(I)
               JAMMER2(I)=HOLD2(I)
320         CONTINUE
         ENDIF
C
199   CONTINUE
C
900   FORMAT(I5)
901   FORMAT(F15.8)
      PRINT*,'LAST MSE VALUE=',MSE,'DB'
C
C  PRINT CONVERGED (HOPEFULLY) WEIGHT VALUES
      OPEN(7,FILE='WEIGHTS.DAT')
      WRITE(7,900) L
      DO 500 I=0,2*L-1
         WRITE(7,901) W(I)
500   CONTINUE
C
      STOP
      END
```

```
      SUBROUTINE RANDOM(X,N,SEED)
C
C ****************************************************************
C     THIS WILL GENERATE A UNIFORMLY DISTRIBUTED SEQUENCE
C     OF LENGTH N.
C ****************************************************************
C
      REAL*4 X(1025)
      INTEGER SEED,N
C
      DO 10 I=1,N
         SEED=2045*SEED+1
         SEED=SEED-(SEED/1048576)*1048576
         X(I)=REAL(SEED+1)/1048577.0
10    CONTINUE
      RETURN
      END
      SUBROUTINE FILTER(NOISE,OUTPT,REG,N)
C
C *****************************************************************
C          THIS IS A FOURTH ORDER BUTTERWORTH FILTER USED TO
C      COLOR WHITE NOISE FOR THE PURPOSE OF GENERATING A BROADBAND
C      SIGNAL. CENTER FREQ=PI/2. BW=PI/2
C *****************************************************************
C
      REAL*8 REG(0:9),A0,B1,B2,OUTPT(1250),NOISE(1250)
      INTEGER I,N,J
C  FILTER COEFFICIENTS
      A0=.292893219
      B1=.3101797215
      B2=SQRT(2.)-1.
C
      DO 10 I=1,N
C   SHIFT REGISTERS
         REG(2)=REG(1)

         REG(1)=REG(0)
         REG(5)=REG(4)
         REG(4)=REG(3)
         REG(8)=REG(7)
         REG(7)=REG(6)
C  1ST SECTION
         REG(0)=NOISE(I)*A0
         REG(3)=-REG(2)+B1*REG(4)-B2*REG(5)-REG(0)
C  2ND STAGE
         REG(6)=REG(3)-REG(5)-B1*REG(7)-B2*REG(8)
         OUTPT(I)=REG(6)
10    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INTERP(SEQN,STWN,H,SIZE,M)
C
C *****************************************************************
C          THIS IS A SYMMETRICAL SAMPLED SINC INTERPOLATOR
C      WEIGHTED BY A HAMMING WINDOW. AN INPUT SEQN. OF
C      LENGTH=SIZE IS INTERPOLATED TO A NEW SEQUENCE ACCORDING TO
C      'DELAY'.  SIZE-2M IS THE NUMBER OF PTS. RETURNED, WHERE
C      2M+1 IS THE INTERPOLATOR LENGTH.
C *****************************************************************
C
      REAL*8 SEQN(1250),H(0:64),STWN(1250)
      INTEGER I,J,M,SIZE,TWOM
      TWOM=2*M
      PI=3.141592654
C
      DO 40 J=0,SIZE-TWOM-1
         STWN(J+1)=0.
         DO 50 I=0,TWOM
            STWN(J+1)=H(I)*SEQN(J+1+TWOM-I)+STWN(J+1)
50       CONTINUE
40    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FHT(X,Y,N)
C ***************************************************
C  computes the hadamard transform of x. coefs are
C  sequency ordered.  N=2**NU. SEE REF  10!
C ***************************************************
C
      REAL*8 X(128),Y(128)
C
      NU=NINT(LOG10(REAL(N))/LOG10(2.))
      N1=N-1
      N2=N/2
      NU1=NU-1
      N4=N/4
      DO 5 I=1,N
         Y(I)=X(I)
5     CONTINUE
      J=1

      DO 13 I=1,N1
         IF (I.GE.J) GOTO 12
            T=Y(J)
            Y(J)=Y(I)
            Y(I)=T
12          K=N2
10       IF (J.LE.K) GOTO 11
            J=J-K
            K=K/2
            GO TO 10
11       J=J+K
13    CONTINUE
      DO 40 I=1,NU
         IA=2**I
         IB=IA/2
         DO 41 J=1,IB
            DO 42 K=J,N,IA
               IC=K+IB
               Z=Y(IC)
               Y(IC)=Y(K)-Z
               Y(K)=Y(K)+Z
42          CONTINUE
41       CONTINUE
40    CONTINUE
      DO 60 I=1,NU1
         IGA=2**(I-1)
         DO 70 L=1,IGA
            DO 71 J=1,N4
               IY=N2+J+(L-1)*N/IGA
               IX=IY+N4
               Z=Y(IX)
               Y(IX)=Y(IY)
               Y(IY)=Z
71          CONTINUE
70       CONTINUE
         N2=N2/2
         N4=N4/2
60    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DCT(IN,OUT,NUM)
C
C*****************************************************************
C  DISCRETE COSINE TRANSFORM WHICH UTILIZES A
C  SIMPLE FFT PROGRAM WHICH USES DECIMATION IN TIME.
C*****************************************************************
C
      COMPLEX*8 Y(1024),U,W,T
      REAL*8 IN(512),OUT(512),PI,ARG
      INTEGER N,M,NUM
      PI=3.14159265358979
      DO 3 I=1,NUM
         Y(I)=CMPLX(IN(I),0.)
3     CONTINUE
      DO 4 I=NUM+1,2*NUM
         Y(I)=(0.,0.)

4     CONTINUE
      N=2*NUM
      M=ALOG(FLOAT(N))/ALOG(2.)+.1
         NV2=N/2
         NM1=N-1
         J=1
         DO 40 I=1,NM1
            IF (I.GE.J) GOTO 10
               T=Y(J)
               Y(J)=Y(I)
               Y(I)=T
10          K=NV2
20          IF (K.GE.J) GO TO 30
               J=J-K
               K=K/2
               GO TO 20
30             J=J+K
40       CONTINUE
       DO 70 L=1,M
         LE=2**L
         LE1=LE/2
         U=(1.,0.)
         W=CMPLX(DCOS(PI/FLOAT(LE1)),-DSIN(PI/FLOAT(LE1
         DO 60 J=1,LE1
            DO 50 I=J,N,LE
               IP=I+LE1
               T=U*Y(IP)
               Y(IP)=Y(I)-T
               Y(I)=Y(I)+T
50          CONTINUE
         U=U*W
60       CONTINUE
70    CONTINUE
      DO 80 I=1,NUM
         ARG=PI*REAL(I-1)/REAL(N)
         OUT(I)=2.*REAL(Y(I)*CMPLX(DCOS(ARG),-DSIN(ARG
80    CONTINUE
      OUT(1)=OUT(1)/SQRT(2.)
      RETURN
       END
```

END

5-87

DTIC